INVESTIGATION
INTO THE
WAFER-SCALE INTEGRATION
OF
FINE-GRAIN PARALLEL PROCESSING
COMPUTER SYSTEMS

A thesis submitted for the degree of Doctor of Philosophy

by

Simon Richard Jones

Department of Electrical Engineering and Electronics

Brunel University

November 1986

ABSTRACT

This thesis investigates the potential of wafer-scale integration (WSI) for the implementation of low-cost fine-grain parallel processing computer systems.

As WSI is a relatively new subject, there was little work on which to base investigations. Indeed, most WSI architectures existed only as untried and sometimes vague proposals. Accordingly, the research strategy approached this problem by identifying a representative WSI structure and architecture on which to base investigations.

An analysis of architectural proposals identified associative memory to be general purpose parallel processing component used in a wide range of WSI architectures. Furthermore, this analysis provided a set of WSI-level design requirements to evaluate the suitability of different architectures as research vehicles. The WSI-ASP (WASP) device, which has a large associative memory as its main component is shown to meet these requirements and hence was chosen as the research vehicle.

Consequently, this thesis addresses WSI potential through an in-depth investigation into the feasibility of implementing a large associative memory for the WASP device that meets the demanding technological constraints of WSI.

Overall, the thesis concludes that WSI offers significant potential for the implementation of low-cost fine-grain parallel processing computer systems. However, due to the dual constraints of thermal management and the area required for the power distribution network, power density is a major design constraint in WSI. Indeed, it is

shown that WSI power densities need to be an order of magnitude lower than VLSI power densities. The thesis demonstrates that for associative memories at least, VLSI designs are unsuited to implementation in WSI. Rather, it is shown that WSI circuits must be closely matched to the operational environment to assure suitable power densities. These circuits are significantly larger than their VLSI equivalents. Nonetheless, the thesis demonstrates that by concentrating on the most power intensive circuits, it is possible to achieve acceptable power densities with only a modest increase in area overheads.

ACKNOWLEDGEMENTS

Firstly, I wish to thank Professor R.M. Lea, for his continual support and guidance throughout this project. Without his consistent encouragement for the development of my work, this thesis would not have been possible. Furthermore, my colleagues on the SCAPE and WASP projects, namely Mohamed Abdelrazik, Hamid Bolouri, Stephen Hedge, Ian Jalowiecki, Constantine Katolean, Ray McKirdy and Ken Warren are gratefully acknowledged for their lively discussions, which undoubtedly have influenced my views on the potential of WSI. The academic and industrial collaborators on the Alvey-funded WSI project (VLSI/ARCH/073) are thanked for their encouragement. The support of my parents and Ms Susan Gough during this period, has been invaluable.

Finally, the financial assistance of the SERC is gratefully acknowledged.

CONTENTS

CHAPTER 1. INTRODUCTION

1.1 Perspective

In recent years, the field of information technology has seen a rapid expansion in application areas which demand low-cost high-speed computer systems. These areas include,

(1) Image processing: including high-speed computer graphics, medical electronics (eg. bodyscanners, X-rays) and the wide fields of industrial robotics and artificial vision.

(2) Signal processing: applications involving the analysis and interpretation of wide range of signals including radar, telecommunications and speech processing.

(3) Intelligent knowledge-based systems (IKBS): to support an 'expert systems' approach to a wide range of problems (eg. medical diagnosis, traffic management systems).

(4) Fifth-generation high-level language support (eg. Prolog).

These applications are wide ranging and involve a variety of computationally intensive tasks. For example, a simple image processing binarised edge detecting algorithm, using a 3 x 3 filter on a 512 x 512 pixel image, operating at television rates (25 frames per second), involves in excess of 350 million operations per second [1]. A processing rate approached only by the most powerful of mainframes. For more sophisticated operations, the processing speed required could easily exceed 10^9 operations per second. Currently, it is highly cost-ineffective to control, for example, a simple

industrial robot making decisions based on visual input, with a computer system of such power.

However, the potential market volume in these application areas (particularly computer graphics and industrial robotics) is such as to have stimulated much interest in the design of parallel processing computer systems for low-cost high-speed data processing.

1.2 Computational parallelism

It is widely accepted that for general purpose applications, computational power does not increase linearly with the number of processor elements. Rather, since a proportion of the parallel computer's work-load must be executed sequentially, the 'natural' parallelism (viz. the potential for simultaneous execution of independent processes) reduces the utilisation of the 'N' processors in the system to $N/\log_2 N$.

This is the ideal case. However, since the particular structure of the parallel processing computer may not exactly match the 'natural' parallelism (eg. a 2-D array of processors attempting to solve a 3-D problem, such as weather forecasting) a further reduction in processor utilisation occurs, reducing the 'applied' parallel processing power to $\log_2 N$.

This difference between natural and applied parallel processing power is shown in Figure 1-1. These characteristic curves leave 2 main options available for parallel processor systems designers, namely

(1) Use a few very powerful processors (medium-grain parallelism) to stay close to the origin of the graph and hence the natural parallelism curve.

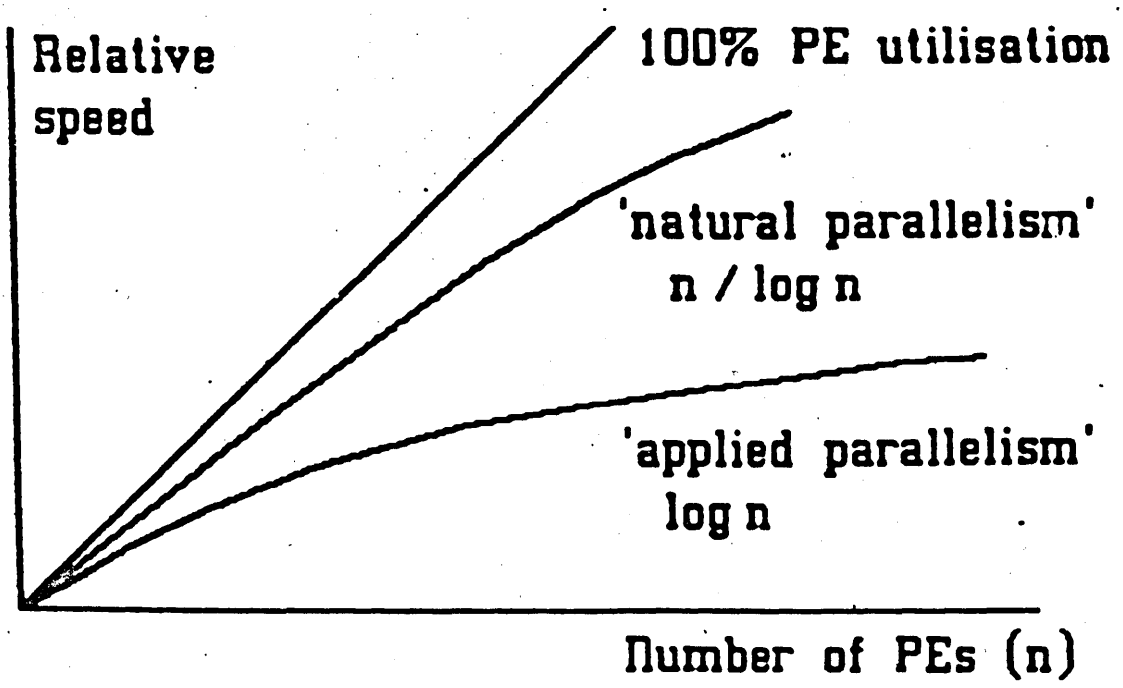


Figure 1-1. Applied and natural parallelism.

(2) Use many small and cheap processors (fine-grain parallelism) to reduce the cost of each processor. Furthermore, make the architectural organisation flexible and programmable to more closely match the parallelism in the problem and hence get closer to the natural parallelism curve.

There are merits with both approaches, but perhaps the major advantage of the fine-grain parallelism approach is that it is better suited to VLSI implementation.

1.3 VLSI parallel processing chip architectures

Recent years have seen significant progress in the field of integrated circuit (IC) design and fabrication, such that computer systems engineers are radically re-appraising their ideas on computer architecture to make use of the cost and performance advantages of VLSI. While VLSI can offer cost and performance advantages to conventional Von Neumann architectures, and, indeed, 32-bit microprocessors and 1-Mbit RAMs are now available as a result of VLSI technology, it is widely thought that fine-grain parallel processor chip architectures have the potential to achieve significantly greater cost and performance benefits from VLSI.

Fine-grain parallel processing chip architectures, comprise a regularly connected array of many simple flexible processing elements (PEs) operating in parallel, to achieve high-speed data processing. Their regularity, iterability and nearest neighbour communications make them well suited to the technological constraints of VLSI. Such VLSI fine-grain parallel processor chips are cost-effective building blocks for the construction of SIMD processor arrays, which attached

to a conventional slow to medium speed host computer, provide the flexible high-speed processing rates demanded by the applications previously discussed, at a cost significantly lower than that of an equally powerful conventional (Von Neumann) computer system.

A range of such chip architectures, incorporating a few tens to a few hundreds of processing elements are currently under design or development. Examples include,

(1) The 8-PE cellular logic image processor (CLIP) [2] and the massively parallel processor (MPP) [3] chips targetted for image processing applications.

(2) The 16-PE ICL distributed array processor (DAP) [4] chip for general array processing applications.

(3) The 32-PE GEC rectangular image and data (GRID) [5] chip. Again, primarily targetted at image processing applications.

(4) The 64-PE adaptive array processor (AAP) [6] chip aimed at image processing, but reported to be applied to CAD applications.

(5) The 72-PE geometric and arithmetic parallel processor (GAPP) [7] chip currently available from NCR and applied to a wide range of computationally intensive problems.

(6) The 256-PE single chip array processing element (SCAPE) chip [8,9], developed at Brunel University based on the associative string processor (ASP) architecture [10,11]. The SCAPE chip is targetted at the cost-effective support of image and signal processing applications.

(7) The 256-PE single chip relational information processing element (SCRIPT) chip [12]. SCRIPT is also developed at Brunel University and like SCAPE, is based on the ASP architecture. SCRIPT is optimised for the support of text processing, database management systems and IKBS applications.

Table 1-1. summarises the PE density and pinout for these chip architectures.

Name	PE density (PE/chip)	Pinout
CLIP [2]	8	40
MPP [3]	8	52
DAP [4]	16	176
GRID [5]	32	128
AAP [6]	64	152
GAAP [7]	72	88
SCAPE [8]	256	68
SCRIPT [12]	256	100

Table 1-1. PE density and pinout.

It can be seen from Table 1-1., that there is a wide range in PE density and pinout between the various architectures. To some extent this can be contributed to the fabrication technology used. However, it is interesting to note the wide margin in PE packing density between the ASP-based chips (SCAPE, SCRIPT) and the other architectures, together with the high PE-to-pin ratio of the ASP architectures.

The implications of the difference in PE packing density is better appreciated when the number of chips to realise a particular array size is compared (Figure 1-2.). For example, consider the implementation of an 8192-PE array (eg. 1024 x 8, 512 x 16, 256 x 32, 128 x 64). The 8-PE chips require over a thousand processor chips to realise an 8192-PE array. The 72-PE GAPP chip still requires over a hundred chips. Even with the use of the 256-PE SCAPE and SCRIPT chips, 32 processor chips are required. Such systems may be bulky and require many printed circuit boards, resulting in expensive, difficult to engineer for high-speed operation and potentially unreliable implementations. Consequently, many researchers have been investigating ways of improving PE packing density in an effort to increase overall system cost-effectiveness.

Increasing the number of PEs per chip (namely, PE packing density) increases die size, which in turn significantly affects chip yield. Currently, with chips much larger than 1 cm^2, yields are often less than 1%. If fault-tolerance (viz. the ability to configure a working subsystem from a partially faulty system) and redundancy (viz. the ability to incorporate spare PEs) can be added to the architecture, then the yield limit on die size is avoided, allowing a greater number of PEs to be integrated on a ULSI (Ultra-Large Scale Integration) chip. The ultimate in PE integration would be a WSI (Wafer-Scale Integration) [13] implementation of the architecture, integrating a fine-grain parallel processor comprising potentially, many thousands of PEs or even a complete parallel processing system, on a single undiced silicon slice.

As the following section shows, WSI offers many potential benefits for the implementation of parallel processing computer systems.

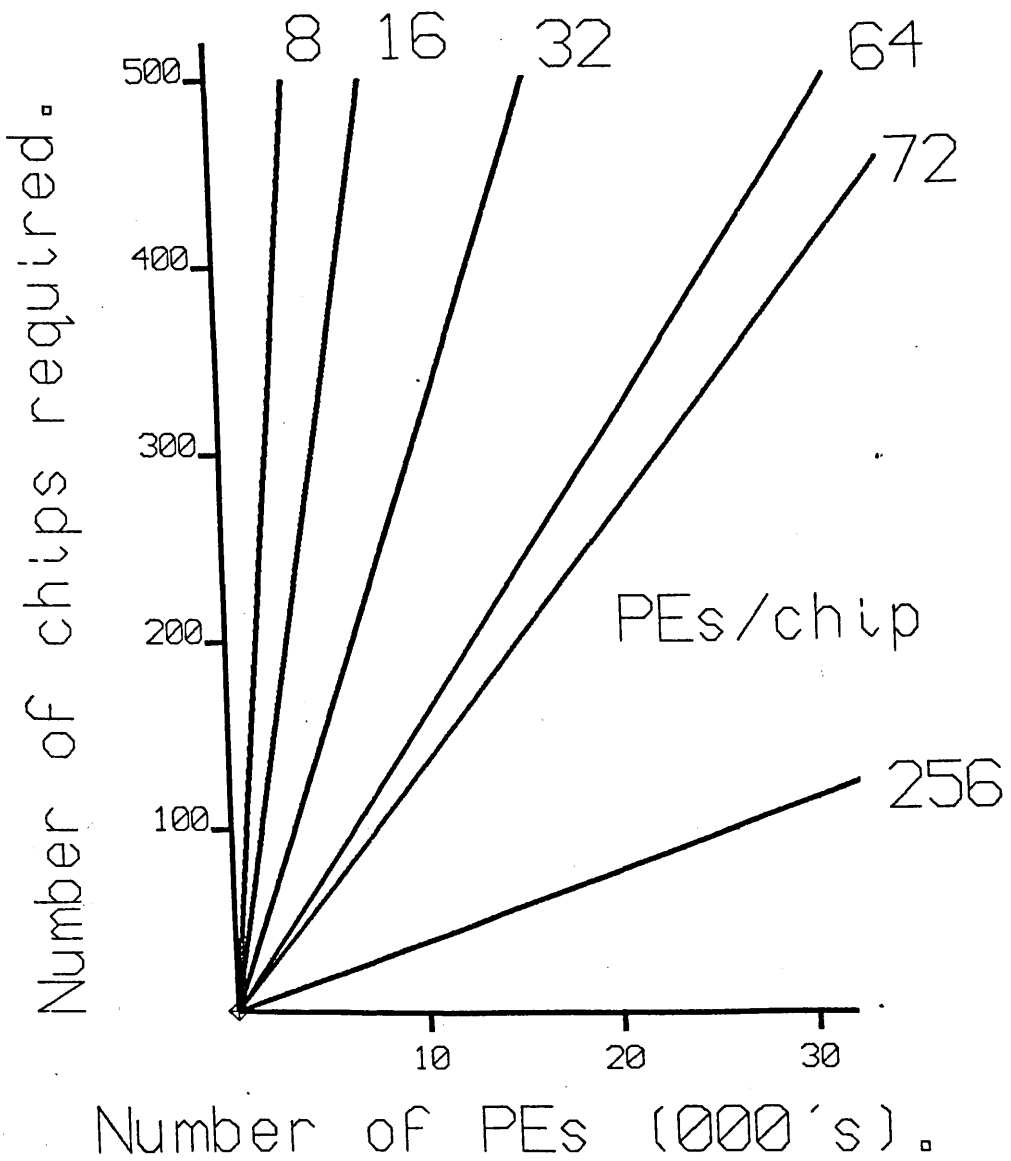


Figure 1-2. Chip count vs PE array size

1.4. Advantages of wafer-scale integration

A workshop at Brunel University [13], identified WSI as providing the following main advantages for computer systems integration

(1) Cost

(a) Silicon-efficiency, namely the amount of silicon that needs to be processed to realise a particular system. The low yield of VLSI chips means that a large amount of silicon must be processed in order to realise one working device. In WSI, the use of fault-tolerance and redundancy means that each module has a much higher yield and thus less silicon needs to be processed to realise the WSI-based system.

(b) Packaging advantages comprise two main areas

(i) Materials: the reduction in the number of packages decreases chip and board materials cost.

(ii) Assembly: the reduction in the number of packages reduces assembly related costs.

(c) Testing: WSI allows the three levels of VLSI testing (viz. wafer probing, chip test and system test) to be compressed into one single test. Although, limited wafer probing may still be needed to check process parameters.

(2) Speed: WSI offers two main speed improvements

(a) Clock speed is improved through the potential to incorporate on-wafer clock multiplication, to speed up

system operation over a wider area of functionality than is economically feasible in VLSI. Furthermore, the shorter communication pathways in WSI should also assist in the support of higher clock speeds.

(b) Improved communications bandwidth is likely to be a major advantage of WSI. Communication on silicon is essentially free, whereas communication off silicon is severely restricted by the system economics of PCB implementation, pack to pack and board to board communications. Thus, module to module communication in WSI need no longer be constrained by pinout limitations (cf. chip to chip communication), but rather, can use high-density metal tracking to achieve ultra-wide bandwidth communications.

(3) Power: WSI offers potential power reductions as a consequence of the greatly reduced inter-module capacitances and the significant reduction in the number of power-intensive pad drivers.

(4) Size and weight: clearly, WSI implementation offers the potential to reduce system size and weight significantly by incorporating many boards of devices onto a single silicon slice.

(5) Reliability is improved as the fault-prone package to board connections are replaced by reliable metal tracks on silicon. Furthermore, the use of redundancy and reconfiguration may permit 'graceful degradation' to be designed in, improving system reliability further still.

Clearly, WSI offers many potential benefits over VLSI for the

realisation of low-cost high-speed computer systems. However, successful use of WSI requires significant technological problems to be overcome.

1.5 Problem areas in wafer-scale integration

The same workshop at Brunel University [13] also identified the key problem areas likely to be faced in the implementation of computer systems in WSI

(1) Fault-tolerance, is perhaps the most obvious constraint faced in the design of WSI computer systems. The fault-tolerance scheme must be suited to the particular architecture and cope with the expected range of defects.

(2) Testing, also poses major problems. In VLSI the test costs due to the poor controllability and observability of complex devices are well known. WSI, with a potential two orders of magnitude increase in device complexity could well prove to have unacceptably high test overheads for cost-effective computer systems implementation.

(3) Electrical design constraints, while less obvious perhaps, than fault-tolerance and testability, also poses major design constraints. These design constraints may be broken down into two main areas

(a) Power distribution present two main problems in WSI: firstly, power rails are globally distributed signals and a single fault in the power distribution network could make an entire wafer unuseable. Secondly, the high power

density of WSI circuits, resulting from the increase in PE packing density and the potential for increased clock speeds, combined with the fact that the system power distribution network has now moved from low resistance thick Copper tracks to high resistance thin Aluminium lines, suggests that the area of the power distribution network in WSI may be significantly greater than in VLSI.

(b) Clock and signal distribution, face the same problem as power distribution in that they are global signals and hence a fault in these signals may have a catastrophic effect on the system. In addition, they face the problem of the RC delay constant in wafer length tracking, significantly increasing signal delays and hence having the potential to reduce system performance.

(4) Physical design constraints, stem from the packaging of such a large silicon area. The functions of a wafer-scale package are identical to its VLSI counterpart: thermal management, connectivity and environmental protection. However, problems such as voids between package and wafer, thermal 'hot spots' on the wafer and the thermal mismatch between package and wafer, are aggravated when contemplating packaging a piece of silicon several inches in diameter. To date, no commercial WSI package is available and hence WSI implementation may involve costly package development.

1.6 Research objectives

The previous section has shown the existence of an applications requirement for low-cost high speed processing. Furthermore, fine-

grain parallel processor chip architectures have been suggested to be promising structures for the construction of such systems. The use of WSI has been argued to have the potential to significantly increase the cost-effectiveness of such computer systems.

It is the objective of this thesis to investigate the potential of WSI for the implementation of low-cost, fine-grain parallel processing computer systems. This thesis proposes to explore the following,

(1) Design constraints of WSI and the implications these constraints have on the engineering of parallel processor systems in WSI.

(2) Suitable design techniques for implementing WSI parallel processor computer systems.

1.7 Research Strategy

While the use of WSI was originally proposed over 20 years ago, until recently interest in it has been sporadic. As a result, there is little work available on which to base research investigations. Indeed, despite a great deal of interest in the last few years, current WSI architectures exist only as design proposals, often based on vague and largely untried architectures. These factors raised questions required to be addressed by the research strategy, namely

(1) How to investigate WSI design issues in-depth on the basis of only proposed and untried or vague architectures?

(2) How to ensure that the structures investigated be representative of fine-grain parallel processors in general?

(3) What architecture should be used as a research vehicle to extrapolate from?

In order to address these questions, the following two-part strategy was adopted.

Firstly, an architectural component representative of fine-grain parallel processing WSI computer systems was identified. The thesis then concentrates on investigating the implementation of this component in WSI. In order to prevent the number of design options from proliferating meaninglessly and to provide realistic constraints and targets, the second part of the strategy analyses WSI architectural proposals in order to identify a set of WSI-level design requirements. These requirements would then be used to identify a promising WSI architectural proposal to act as research vehicle for investigations.

Many different structures could have been chosen to be the architectural component for in-depth investigations, but associative memory has been selected for the following reasons.

(1) An associative memory is a well-defined fine-grain parallel processing structure.

(2) The analysis of WSI research work indicated that several of the WSI-based computer systems that have been proposed to date make extensive use of associative memory.

(3) As is demonstrated in this thesis, an associative memory is a flexible structure capable of efficiently supporting a wide range of numerical and non-numerical computational tasks.

This still left the problem of identifying a suitable architecture to use as a research vehicle. The proposed WSI-ASP (WASP) architecture, which has a large associative memory as its main parallel processing component, has been evaluated against the WSI-level design requirments and it is shown that WASP satisfies these design criteria and hence was selected as the research vehicle.

Such a WASP device, could integrate potentially several thousand PEs on a single silicon slice. A WASP device could thereby significantly improve the cost-effectiveness of high-speed computer systems for the support of a wide range of applications in image and signal processing, intelligent knowledge-based systems and 5^{th} generation high-level language support.

Consequently, the research strategy of this thesis is to address the potential of WSI through a set of specific investigations into the implementation of an associative memory for the WASP device.

A variant of the WASP architecture, optimised for the cost-effective support of image processing applications, known as WASP IP is used as the specific research vehicle. The WASP IP device has been chosen for the following reasons.

(1) It is the WASP variant that has been furthest developed to date.

(2) The WASP IP device has a well established applications and systems requirement.

(3) It has a VLSI equivalent, thus allowing direct comparisons to be made between the VLSI and WSI implementation of identical systems.

1.8 Structure of the thesis

This thesis is organised into nine chapters, of which, this is the first.

The second chapter critically appraises the past work on WSI parallel processor architectures to evaluate previous research work and hence to expose underlying WSI-level design requirements for the research vehicle. Associative memory is identified as a component widely used in several architectural proposals.

Chapter three, introduces associative memories and associative processing and demonstrates their suitability for a wide range of numerical and non-numerical computational tasks.

Chapter four, describes a particular associative processor, the WASP device, focussing on the WASP image processing variant. The results from a comparative evaluation of proposed VLSI and WSI implementations of an ASP-based image processing module are reported to further clarify the potential advantages to be gained from using WSI as an implementation technology for ASPs. The proposed WASP device is compared both with the WSI-level design requirements identified in Chapter 2 and also against other proposed WSI architectures. These comparison suggest that indeed, from a broad perspective, WASP does have the characteristics indicative of a design suitable for cost-effective WSI implementation and hence is considered to be a suitable research vehicle.

The purpose of Chapter five is to define the investigation strategy used to investigate in-depth, the implementation of a suitable

associative memory for WASP that meets the technological constraints of WSI. The relevant questions that need to be answered are identified and experiments are proposed to address these questions. The experimental procedures and treatment of the experimental data are described.

The sixth chapter is concerned with evaluating the electrical, physical and control characteristics of associative memories. It identifies the WASP associative memory requirements and proposes a range of associative memories for further investigation. These designs are laid out and their electrical properties simulated using SPICE. The associative memory control requirements are identifed and the overheads involved in the use of these designs discussed. Following this, the latter part of the sixth chapter is concerned with evaluating the performance of these designs within the WASP architectural and operational environment. Finally, details of the selection and use of one of these designs in the SCAPE chip are reported.

The seventh chapter investigates the yield and harvest implications of using the different associative memory designs in the image processing WASP device.

The eighth chapter aims to expose the implications that the use of different associative memory designs has on the characteristics of the WASP device. This investigation evaluates the suitability of the different associative memory designs for meeting the electrical and physical constraints of WSI, together with the influence these designs have on PE packing density.

Chapter nine draws together the experimental results and arrives at the main conclusions of the thesis as to the potential of WSI for the

implementation of fine-grain parallel processor computer architectures. The thesis is criticised and areas where future work is needed are identified.

CHAPTER 2. REVIEW OF WSI PARALLEL PROCESSOR ARCHITECTURES

2.1 Objectives

The objectives of this chapter are as follows,

(1) To critically appraise the past work on WSI, focussing on WSI parallel processor architectures.

(2) To use this appraisal to analyse and expose underlying characteristics of WSI architectures and hence to identify WSI-level design requirements for the research vehicle.

(3) Identify a parallel processing architectural component common to a range of proposals, suitable for being used in a wide range of applications.

2.2 WSI overview

The first recorded reference in the literature to WSI is by Sack, Lyman and Chang [14]. They propose an array of very simple logic cells on a wafer being probe tested. The functional cells are then interconnected via a customised metallisation layer. Later, this technique was to become known as discretionary wiring. Sack, Lyman and Chang, envisaged being able to integrate an entire ALU on a single wafer and thus were understandably enthusiastic in 1964. However, no further references to this work are recorded in the literature.

In [15], Petritz discusses the advantages of wafer-scale integration, or as it was then called, full-slice technology. Petritz argued

strongly that the full potential for semiconductor electronics will be achieved only when the entire slice constitutes the packaged product.

In the late 1960's, a research group at Texas instruments investigated the feasibility of using discretionary wiring as a means of implementing WSI circuits. The experiments were reasonably successful. Indeed, prototype wafers were produced [16]. However, the cost of customising a mask for each wafer, together with the potential for the extra metal mask to damage previously good cells, stopped the project developing further.

In an attempt to reduce the cost of customising masks, Calhoun [17,18] proposed a technique known as pad relocation. The technique effectively trades cell redundancy for using the same discretionary mask among many wafers.

By analysis of the defect pattern on a range of wafers, a core set of cells are identified that are assumed to work. A mask is designed to connect these cells together. For those wafers (the majority) that have this core set of working cells, the same discretionary mask can be used. For those wafers, with some of the core cells being faulty, then the pads of nearby good cells may be relocated (by an extra small and simple mask) to the core cells. Using this technological-led approach, prototype pad relocated wafers were produced but the project was eventually dropped.

From the early to mid-1970s, there was little work published on WSI, However, between 1976 and 1977 three papers were published which did much to reawake interest in WSI.

The first was the associative linear array processor (ALAP) [19],

closely followed by Manning's research into cellular arrays [20] and Aubusson and Catt's work on integrating semiconductor memory [21]. Indeed, it was Aubusson who coined the term 'wafer-scale integration'. Since this date, there has been a significant renewal of interest in WSI. Many researchers have been looking at WSI memory structures, with ROM [22] and RAM [23] devices being developed in the NTT laboratories in Japan. In the UK., Sinclair Research has developed a memory [24] based on work at Burroughs [25]. Elmer, et al [26] implemented a large-area, fault-tolerant CCD memory. Barsuhn of IBM Germany, [27] developed a fault-tolerant 8 K-words x 9 bit WSI memory for use as the writeable control store in an IBM mainframe. In the USA, Shaver [28,29] has designed a WSI EPROM that uses E-beam technology for configuration.

WSI has also been proposed as a technology for implementing conventional systems, [30,31], the most notable being Trilogy's attempt [31] to implement an IBM 3081 processor equivalent on a single wafer. Trilogy dropped WSI due to a combination of troubles with their advanced process technology (6 metal layers and a copper power bus) and the difficulty in adding sufficient fault-tolerance to an irregular architecture.

Others (eg. Johnson [32]), have approached WSI from a packaging viewpoint and used a silicon slice as a low-cost substrate, by down-bonding VLSI chips onto a wafer. This provides the high bandwidth of WSI without the necessity for fault-tolerance to be included and as such, would appear to be an approach with merit; certainly it may prove to be a useful intermediate step to large area integration.

The most noticeable area where there has been an upsurge in interest,

is in the use of WSI as an enabling technology for the implementation of parallel processor architectures.

The following sections review the major research projects concerned with the wafer-scale integration of parallel processors. While there has been extensive treatment in the literature on the mathematical analysis of fault-tolerance and WSI, the following sections address those architectures close to 'real-life' design and development.

2.3 Special purpose architectures

These represent architectures that perform a fixed function (ie. not programmable).

2.3.1 Restructurable VLSI

At Massachusetts Institute of Technology, a large research program has been undertaken into the use of wafer-scale integration for application-specific uses. Termed RVLSI, (restructurable VLSI) [33], it uses a laser 'zapping' technology for configuration.

An RVLSI wafer typically consists of an array of a few different cell types of MSI/LSI complexity. Many cell designs incorporate redundancy to increase yield. The cells lie as islands within a manhattan mesh of wires that pass orthogonally to each other on seperate metal layers. After fabrication, the cells are probe tested. Laser cutting and welding of the mesh of wires is then performed to create the desired connectivity. This laser zapping is essentially irreversible (although extra connections could be made at a later stage) and is reported to be reliable [34].

To date, several RVLSI wafers have been created. The first was

reported by Garverick [35] and is a 16-point 16 MHz FFT processor. Later, a digital integrator wafer was designed [33]. Rhodes in [36], reports the application of the RVLSI technology to a constant false alarm rate filter, for use in radar signal processing, that has the same multiplier-accumulator basis as in the FFT processor. Also, a dynamic time warping processor has been fabricated using RVLSI to support continuous speech recognition.

Clearly, the RVLSI technology has been shown to be feasible. Furthermore, its use is by no means restricted to application specific wafers. the same principles could be applied to a general purpose processor architecture. Although, in order to support fault-tolerance, a significant degree of circuit regularity would probably be neccessary.

2.3.2 Renneslaer Polytechnic wafers

This approach involves the use of discretionary wiring [37], to realise a WSI system. An array of (potentially many) different cell types are probed to distinguish good and faulty circuits. A CAD suite is used to define a discretionary interconnect mask. An interesting aspect of the Renneslaer approach is their use of thick metal lines on the interconnect mask. These lines behave as lossy LC transmission lines and hence avoids the slow RC lines on conventional ICs, which could otherwise significantly slow down cell to cell communication. A characteristic of the Renneslaer approach is its heavy reliance on automatic routing and placement for the cells. To date, two wafers are under development: an FFT processor and a multiprocessor database machine.

2.3.3 Systolic arrays

Their regularity, nearest-neighbour communication and parallel operation, make systolic arrays well suited to VLSI implementation. Consequently, there has been strong interest in the use of WSI as a medium for the implementation of very large systolic arrays. Although, with the exception of Moore's work, this has tended to be more theoretical than practical.

2.3.3.1 Moore's systolic arrays

Moore, has investigated the implementation of fault-tolerant VLSI and WSI systolic arrays in some detail. His work addresses the implementation of a fault-tolerant network of one and two-dimensional systolic arrays. In [38], Moore and Day investigate techniques for improving the yield of one-dimensional systolic arrays of multiply-accumulate cells. Assuming that standard IC technology is used (ie. transistors are used for switching), a range of techniques are evaluated, included skipping 'n' bad cells, multiple skip paths and nearest neighbour connectivity. He concludes that a scheme whereby faulty chips enable their own bypass, gives the highest usage of cells. He suggests that this will allow systolic chips up to 8 times the present size to be successfully fabricated.

In [39] Moore and Mahat look at techniques for providing fault-tolerant communications for 2-D processors (which need not neccessarily be systolic). They demonstrate that in attempting to map a 16 x 16 array of good cells, out of an array of partially faulty cells, a cell yield of 90% is needed to achieve a 95% wafer yield, with 43% cell redundancy. However, if cell yield slips below this,

then the cell overhead increases rapidly (eg. 65% cell yield gives a 96.5% wafer yield but at a cost of 170% cell redundancy). Overall, the results of Moore's work suggest that using standard transistor switching and nearest-neighbour connectivity, one-dimensional arrays are easiest to make fault-tolerant. However, provided a high cell yield can be assured, two-dimensional arrays can be cost-effectively supported.

2.3.3.2 Kung and Lam's systolic ring

Kung and Lam [40], have investigated the implementation of fault-tolerant systolic arrays. They concentrate on one-dimensional arrays and introduce the notion of a systolic ring, whereby the output of a systolic array is fed back to its input. Faulty cells are replaced by clocked delay elements, thus preserving synchronicity. The systolic ring offers the potential to maintain functionality but at the cost of reduced performance. Laser configuration is assumed for implementing the fault-tolerance.

2.3.3.3 Leighton and Leiserson's systolic arrays

Leighton and Leiserson [41], have investigated the WSI of systolic arrays. Their work assumes a laser programmed interconnect and focusses on assembling a large systolic system on a wafer. They attempt to minimise interconnect distance. Mathematically verified procedures are developed to link arrays of cells into the desired structures (eg. strings, arrays, trees). Currently this work is restricted to a mathematical analysis.

2.3.3.4 Diogenes

Rosenberg [42,43], has proposed the Diogenes system. Which is a general purpose system based on a bus structure for configuring a set of partially faulty cells into particular structures (eg. one and two-dimensional systolic arrays, trees, hypercubes). Like Leighton and Leiserson's work, the work concentrates on a mathematical treatment of the problem. Although, circuit diagrams to perform the PE connectivity are proposed [43].

2.3.3.5 Summary

Clearly, systolic arrays, and especially one-dimensional systolic arrays due to their amenability to fault-tolerance, appear to be interesting architectures for wafer-scale integration. Unfortunately, their fixed function does restrict their area of application. However, a programmable function systolic array, could well prove to be a cost-effective structure for WSI implementation.

2.4 Array processors

There has been much interest shown in the wafer-scale integration of array processors. An array is a powerful interconnection structure for parallel processing. Furthermore, it can serve as a universal structure into which arbitary data structures can be embedded.

2.4.1 Pipelined elements

A number of researchers have investigated the implementation of pipelined processor architectures in WSI.

2.4.1.1 Darmstadt University FFT wafer

In [44], Schuck and Glesner of the Technische Hochschule, Darmstadt (Germany), investigate the implementation of a pipeline architecture for the support of two-dimensional FFTs. Inter-processor connectivity is supported either through laser programming or electrical switches. The processing element is estimated to comprise 100,000 transistors.

2.4.1.2 Fried's pipelined processor

Fried in [45] investigates the wafer-scale integration of pipelined processors. A general purpose self-testing pipeline element (PE) is used as the basic building block for a range of structures, including arrays. Soft configuration through skipping bad PEs is mainly used, but Fried uses E-beams to disconnect bad chips that pass the self-test routines. The use of specialised PEs for arithmetic and external storage control is proposed to speed up system operation.

2.4.2 Configurable highly parallel (CHiP) processors

Since the early 1980's a parallel processor architecture known as CHiP [46,47] (Configurable Highly parallel Processor) has been developed at Purdue University. Originally, CHiP was designed for VLSI, but the principles of the CHiP system makes it well suited to fault-tolerant implementation. Each PE has a simple arithmetic-oriented instruction set and a 64 byte memory. CHiP processors are arranged in an array and connected to neighbouring PEs via a switch lattice. The CHiP designers were concerned to minimise inter-PE communciation distances and adopted a two-level hierarchical scheme. As with Moores work [39], they identify the requirement for each PE (viz. CHiP processor) to have a high PE yield, in order to achieve a

useable structure (viz. a wafer with consistently short distances between neighbouring good PEs). A hierarchical scheme is used. Each node is considered as a 2 x 2 PE array. By incorporating 12 PEs per node (viz. a 4 x 3 array), and configuring them into a 2 x 2 array, 99% of the nodes are expected to work. The occasional faulty column is bypassed. Thus each CHiP wafer, can be expected to yield sufficent PEs. However, this is achieved at the expense of massive redundancy.

2.4.3 Wafer-scale systolic processor

A development from Snyder's work is the wafer-scale systolic processor (WASP) device reported by Hedlund and Snyder [48]. Confusingly named, since it has more in common with SIMD/MIMD array processors than systolic arrays (and it conflicts with the WSI-ASP device), it uses the same two level hierarchy and programmable routing switches. In [49], details of a prototype WASP chip are given.

2.4.4. Reconfigurable processor array

At the University of Southampton, research is currently being undertaken into the design of the reconfigurable processor array architecture (RPA) [50,51]. The RPA is a fine grain parallel processor, that comprises a set of bit-serial processing elements that can be composed into wider bit-width devices. This means the RPA can be used both for bit-serial and bit-parallel operations. Communication between RPA elements is via a 2-bit 4-way nearest-neighbour communication network. The fault-tolerance mechanism used in the WSI RPA is discussed in [51]. A hierarchical fault-tolerance scheme is used. At the lower levels in the hierarchy (viz. the PEs) a

self-organizing network is used to create working rows of N PEs from a row with (2N + 2) spare PEs. If a row has more failures than spares, then it is bypassed under the control of higher levels and a spare row swapped in. These rows are then configured into the desired sub-array. This approach therefore shares some similarity with Hedlund and Snyder's approach of high basic unit yield (viz. a row in the RPA) and the occasional bypass of larger groups of elements.

At the higher-levels 'hard' (ie. non-volatile) links are used to configure the RPA system. Currently, the use of split bonding pads and discretionary gold bonds connecting the two halves of the pad together is being investigated as a low-cost technology for hard restructuring.

2.4.5 HYETI

An architecture currently under development at the Institiute for Applied Mathematics (IMAG), Grenoble University (France) is the HYETI processor [52,53] for the Systolimag machine. The approach is to build a two-dimensional array of coarse-grain (eg. microprocessor units) parallel processor elements connected via a programmable array of switches. In [52], E-beam programming is proposed.

Like previous researchers, they recognise the need for high cell yields. The HYETI approach is to build a high yield 16-bit microprocessor-like element out of an array of bit-slices. Faulty slices are bypassed and spare slices switched in. Furthermore, the control PLAs also possess spare row and column redundancy. The objective is then to achieve a high-yielding node with only modest redundancy overheads. Although, this approach requires potentially

complex test and configuration hardware to switch in and out the slices and program the PLAs.

2.4.6 WINNER

Another approach to the construction of two-dimensional arrays on a wafer is the Wafer-Integration using Nearest-Neighbour Electrical Reconfiguration (WINNER) approach, [54,55] developed by Evans at the Royal Signals Research Establishment, Malvern, UK. WINNER, is a self-configuring system. Each PE self-tests, and then transmits this result to its nearest neighbours. Eight-way connectivity is used to improve cell utilisation. Cells connect to each other according to a defined priority list. Thus a cell will refuse to connect to a neighbour if a higher-priority neighbour is available. With N processors, a stable array structure is achieved in O(2N) iterations. The eight-way connectivity appears to improve cell redundancy overheads, since in [55] a 90% cell yield needs less than 200% cell redundancy (ie. less than 200 PEs to assure a 10 x 10 PE array). At lower cell yields, however the cell redundancy grows alarmingly (850% with a 50% PE yield) as PE yield drops.

2.4.7 3-D WSI processor array

In [56], Etchells, Grindberg and Nudd report on a new approach to the implementation of WSI parallel processor systems. In addition to the use of WSI, they also propose the use of third-dimension interconnection (viz. in the vertical plane, through the slice) to connect stacks of wafers together. This 3-D connectivity greatly enhances the throughput of signals between processor elements. The PEs themselves are simple, bit-serial ALUs, operating in an SIMD

manner under external control. No description of the fault-tolerance strategy is given. Connectivity between wafers in the vertical plane is achieved through diffusing aluminium 'bars' through the wafer and linking seperate wafers via microsprings. The authors propose a 10 cubic inch machine of 256 x 256 processors (64 K-PEs) consuming 20 Watts and capable of performing a wide range of image processing algorithms in real time.

Clearly, such a device could offer significant performance benefits. However, the use of 3-D interconnect is not yet an established technology.

2.5 Distributed-logic memories

Distributed logic memories (DLMs), a class of SIMD architectures, comprise a regular, (usually linear) array of cells, each with an associative storage capability together with simple processing and communication logic.

Many researchers have developed the DLM architecture as a parallel processing architecture for WSI since its regular structure, location-independent (content-addressing) capability and linear connectivity make it well suited to simple fault-tolerance strategies.

2.5.1 Associative linear array processor

The associative linear array processor (ALAP) was developed by Finilla and Love [19] at the Hughes Aircraft company. ALAP is a distributed-logic bit-serial associative processor with data storage in shift-registers. After fabrication, good cells are probe tested. A

discretionary metallisation layer is then applied to connect good cells together. An extra fault-isolation mode exists comprising a few instructions realised through high-reliabiliy (viz. replicated) logic. This allows those previously good cells that have been corrupted through the extra metallisation mask stage to be tested and disconnected. This fault-isolation logic bypasses the faulty module. No more than one consecutive faulty module may be skipped. This facility also allows graceful degradation, since some failures can be switched out. ALAP has data and instructions broadcast to all modules. As with all DLM-architecures, the content-addressing (associative) capability of the modules provides rapid pattern matching. In addition, each module has a simple ALU for arithmetic operations. ALAP is targetted to the support of radar signal processing and high-speed arithmetic.

A 2 inch ALAP demonstrator wafer has been fabricated, comprising 112 modules each holding 64 data bits. The linear nature of the ALAP means that only 20 pins are required for the wafer. The ALAP wafer operates at 1 MHz, although Finilla and Love believe that a more advanced process could support clock rates as high as 10 MHz.

2.5.2 Manning's cellular array

In [20] Manning reports his work on computer-maintained cellular arrays. Mannings approach is to create an array of identical, 4-way, nearest neighbour communicating, programmable function cells, that use soft electronic switches to support fault-tolerance. He then investigates techniques for growing different structures (eg. strings, arrays, trees) from this array of partially faulty cells. The advantage of Manning's approach is that the fault-tolerant

strategy is implemented using conventional IC technology (eg. no laser 'zapping' or discretionary wiring). Furthermore, this soft configuration confers the ability to create different structures as the problem demands.

Manning's technique is as follows: a point on the wafer periphery is selected as a start. It is tested and if good instructs its one of its neighbours to open its link, this process is then repeated with the new 'tip' of the arm. When a faulty cell is encountered, the arm is retracted one cell and another neighbour tested. This process is repeated until no more working and accessible cells exist.

Each one of Manning's cells can perform a simple logic function or act as a one-bit memory. Thus it is posible to to create a specific machine by selecting the function that each cell is to perform.

Once the pattern of good and bad cells is identified, then the desired structure can be created by

(a) Opening the desired combination of links to create the interconnection network.

(b) Programming the cell to the desired function (eg. XOR gate, memory cell).

For linear structures, Manning's approach seems quite efficient; with a 90% cell yield over 80% of the cells can be accessed. At lower cell yields, significantly fewer cells can be 'harvested' (viz. the proportion of working cells that can be incorporated into the system) as faulty cells start to make large areas of the wafer inaccessible to nearest-neighbour communication.

Manning also investigated implementing array structures on his network and finds that arrays are less silicon-efficient than strings. This is caused by the need to use many of his cells as routing switches. For example, with a cell yield of 90% just over 10% of useful cells are harvested.

2.5.3 Aubusson and Catt's spiral

In 1978 Aubusson and Catt [21] reported a fault-tolerant technique for WSI. A set of identical cells are fabricated on a wafer. The cells in this investigation were shift registers, although the technique is extendable to other structures (especially, as will be shown, associative architectures), and connected to their 4-way nearest neighbours via electronic switches. A starting cell is selected and tested by external logic, and if functional: selected its leftmost neighbour, opened the links to that cell and then that cell was tested using the initial cell as a transparent link. Thus, a spiral is grown with the cells gradually becoming linked together. When a faulty cell is encountered, the tip of the spiral retracts and the next neighbour tested. If all neighbours are faulty, the spiral retracts by one cell again. Thus, with a simple test and configure strategy (76 gates per connection link are required to perform this function), a long spiral of cells could be formed. This technique offers cost advantages because conventional semiconductor processing could be used. Furthermore, only one basic cell type is used giving low maskmaking costs and making it amenable to step-and-repeat fabrication. This technique is also amenable to graceful degradation, since if a cell fails in operation, all that is required is for the system to be powered down and a new spiral regrown. A TTL demonstrator was designed.

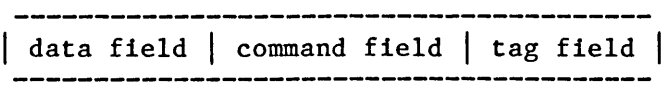
Aubusson and Catt also addressed more practical problems of WSI. Each cell would incorporate a diffused resistance to limit the amount of current a cell could draw, thus limiting the damage a short circuit could cause.

In the same paper, Aubusson and Catt report the result of computer simulations to determine cell harvest with varying cell yields. They found that for an acceptable number of cells to be harvested, with a four-way connection network, cell yields had to be above 65%. Below this yield, increasingly large numbers of cells were inaccessible. In [57] Aubusson, together with Gledhill, investigated the influence different interconnection strategies had on cell yield requirements and harvest. Six and eight-way interconnections were studied, together with the effect on harvest of connecting the edge cells together, to form a unbounded, but finite, toroidal structure. While richer connectivity and toroidal links improved harvest (eg. 80% harvest with a 65% cell yield for an 8-way connected toroid, against 30% for the equivalent yielding 4-way non-toroidal scheme). Much below 65% yield, harvests dropped dramatically (eg. 40% and 8% for the 8-way toroid and 4-way non-toroid discussed previously).

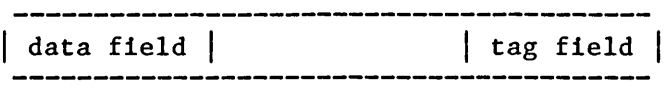
2.5.4 WSI-DLMs

This work of Lea and Sreetharan [58] relates to a development of the Aubusson and Catt spiral approach called Property 1a [59]. The same spiral mechanism is used, but now the cells contain processing logic rather than shift register memory. Furthermore, there are now 2 communication pathways, a fast and a slow line. Data and instructions are passed through the memory in packets. Information on the slow

line is clocked through the cell, whereas information on the fast line flows straight through. Each packet is of the following format:



for instructions and



for data.

When an instruction (on the fast line, for example) meets a data packet (on the slow line, for example) that has the same tag (viz. an associative match) then the command field is applied to the data fields and the result sourced to the downstream cell.

Simulations of this architecture applied to text compression and decompression have been performed and it has been shown [60] that such an architecture can perform disk-rate text compression and decompression.

2.5.5 Cobweb

Cobweb [61,62] is another architecture based on the Aubusson/Catt spiral. Cobweb is targetted at the support of declarative languages. Like the WSI-DLM architecture, the cells now contain processing elements. Unlike the WSI-DLM architecture, once the spiral is established, extra radial links are now opened to allow faster long distance communication. Combinators (viz. a primitive unit comprising a data element and tag information) enter the Cobweb spiral and a tag field in the combinator token identifies the destination. The

combinator is then routed through the radial network until it reaches the right 'ring' and is then passed through the serial pathways to the correct node. In a manner similar to the WSI-DLM architecture, when the tags of two combinators match, the simple logic or arithmetic function specified is performed and a new combinator generated.

As with the other Aubusson/Catt style spiral algorithms, cell yield needs to be above about 65% for reasonable harvests. Kelly and Shute [63] predict a 15% harvest at 65% cell yield but a close to 100% harvest with a 90% yield.

2.6 Analysis

The previous sections have critically appraised a wide range of WSI processor architectures. This section aims to analyse these architectures (summarised in Table 2-1.) with the objective of exposing their underlying characteristics to identify the WSI-level design requirements of the research vehicle.

2.6.1 Architectures

While it may be feasible to implement irregular structures in WSI (although its cost-effectiveness is not clear), the regularity of parallel processor structures does appear to make them better suited to the fundamental problem of fault-tolerance.

There are a few special purpose architectures (eg. the RVLSI work) being designed and developed. However, there exists significant research interest in the implementation of WSI-based parallel processing computer systems for the support of a wide range of

applications. Interestingly, to date there has been no implementation of such a WSI parallel processor system using modern integrated circuit technology and hence the electrical and physical design issues of WSI implementation are as yet, largely unaddressed.

Name	Architecture	Fault-tolerance technology	Interconnection strategy
ALAP	DLM	discretionary	string
Aubusson/ Catt	DLM	standard tech	string
CHiP	processor array	standard tech	hierarchic array
COBWEB	DLM	standard tech	string
HYETI	processor array	laser/E-beam	array
Manning	variable	standard tech	variable
Moore	systolic array	standard tech	array
RPA	processor array	split bondpads	hierarchic array
RVLSI	special-purpose	laser zapping	customised manhattan mesh
Hedlund's WASP	systolic array	standard tech/ laser/E-beam	hierarchic array
WINNER	variable	standard tech	array
WSI-DLM	DLM	standard tech	string

Table 2-1. Summary of WSI architectures.

The reviewed architectures divide into 3 main types, namely

(1) Systolic arrays (eg. Moore's and Kung and Lam's work).

(2) Processor arrays (eg. RPA, HYETI).

(3) Distributed-logic memories (eg. ALAP, WSI-DLM).

The systolic arrays are necessarily restricted in function. Although, a programmable systolic array would be a more flexible device. The processor arrays are programmable, and the use of such arrays to a wide range of numerical problems is relatively well understood. The distributed-logic memory, based on an associative processing approach, is another common architectural proposal and has been applied to both numerical (eg. radar tracking in ALAP) and non-numerical problems (eg. text compression in WSI-DLM).

2.6.2 Fault-tolerance technology

Despite there being a wide range of proposals (Table 2-1), they all have one thing in common: all the architectures need to be able to create a fully-functional sub-system out of a partially faulty wafer. Many architectures have used post-process probing and laser zapping or E-beam programming (eg. RVLSI, HYETI) to configure the desired structure and indeed, there are many advantages to this approach: the customisation need only be done once and the wafer may then be used exactly as any other IC. Furthermore, the flexibility of this customisation is such as to allow a wide range of structures to be configured. Moreover, the use of CAD tools (eg. as in the RVLSI approach) can allow sophisticated routing algorithms to be run to optimise configuration according to the desired criteria (eg. wire length, cell utilisation).

The major constraint of this approach is the requirement to treat every wafer individually. In addition, current laser processing

technology requires that metal tracking must not run over active circuitry, reducing the area available for 'useful' logic.

Furthermore, these approaches require a not insignificant capital investment to provide laser/E-beam equipment and supporting CAD, currently only found in high-volume memory manufacturing centres. Unlike memories, complex processor architectures may require many laser cut and weld operations to be performed. For example, the RVLSI digital integrator in [33] required nearly 2000 tracks to be cut or welded, which is reported to take about 15 mins of laser table time. This may prove too lengthy for large volume applications. However, the use of sophisticated CAD suites and clever architectural and circuit design may reduce the cost of wafer treatment, in a manner analogous to the way in which 'design-for-test' and built-in self-test has made manageable (but not eliminated) VLSI test costs.

For the lowest-cost approaches, the use of standard IC processing (eg. Aubusson/Catt, COBWEB) and hence soft electronic wafer customisation, would seem necessary. This approach has other merits, it is reversible and thus many different structures may be configured (an advantage that Manning makes good use of). Furthermore, in the event of post-process failures it may be possible to reconfigure the system around failures. This provides the possibility of soft-failing systems and graceful degradation to be designed in. A feature that may prove very useful for high-reliability applications, or where the wafer is likely to be inaccessible (eg. satellites). However, it should be noticed that the value of these features is determined by the application and there may be situations where it is cheaper to replace the wafer rather than design-in reliability.

Soft electronic configuration imposes its own penalties. Every time power is removed, the wafer structure is lost. Furthermore, every time the wafer is powered up, the entire wafer must be retested. When testing many complex devices, this could not just take a long time, but also require the test programs to be stored with each device. Again, the use of built-in self-test may reduce this overhead. Also, soft electronic configuration is not ideally suited towards configuring the power distribution network, as the high resistivity of these switches limits the current that can be supplied.

2.6.3 Interconnection strategies

The proposed architectures (Table 2-1.) also use a wide range of interconnection strategies.

When implementing parallel processors, the regularity and iterability of the array and string interconnection strategies seems to make them the natural connectivity structure for WSI.

Intrinsically, the richer connectivity of the array would make it the more desirable structure. However, the need for richer connectivity would seem in some part to be responsible for a major design constraint of the array. Certainly, the research reviewed in this chapter, seems to suggest that to configure an array structure requires a basic element yield of about 90% (eg. Moore, WINNER). Any lower than this, then it gets increasingly difficult to configure a useable structure without excessive cell redundancy and poor cell utilisation (harvest). Although, the use of higher degrees of connectivity (eg. the 8-way connected grid of the WINNER approach) does seem to offer better performance than the simple 4-way connected grid. Nonetheless, the improvement achieved still seems to require

cell yields to be above 80%.

For identical cell yields, the string structure (eg. Aubusson/Catt spiral) with its much simpler connectivity, achieves a higher harvest than the corresponding array structure. However, at yields below 65%, many cells cannot be harvested. Again, the use of higher degrees of connectivity offers many potential paths for the string to be realised. Even so, yields must remain above 50% for a reasonable proportion of cells to be harvested.

To place this in context, consider the data in Table 2-2. A cell of area 25 mm^2, fabricated on a process line of defect density 0.04 d/mm^2 (a 'typical' figure for a modern process line) has a yield of approximately 37% (assuming yield approximates to e^{-da}, where d is defect density and a is cell area). Assuming the same process and yield model, to achieve a cell yield of 65% (adequate for string formation), requires cell size to be about 14 mm^2. For a 90% cell yield, on the same process (adequate for array formations) the cell size must be less than 5 mm^2.

Yield Required	Cell Size (mm^2)
37%	25
65%	14
> 90%	< 5

Table 2-2. Yield required and cell size.
(0.04 d/mm^2 and Poisson statistics).

Such small cell sizes mean that the ratio of 'unproductive' switching logic to 'productive' processing logic may be uncomfortably high (eg. the CHiP switch logic is approximately 1 mm^2).

These uncomfortable statistics require cell yields to be improved if reasonably complex devices are to be implemented. Consequently, hierarchical interconnection strategies are used in some of these architectures. For example, Hedlund implements each wafer-scale systolic processor node as a 3 x 4 array and attempts to map a 2 x 2 array from this. Each node therefore has a very high yield. Consequently, it is easier to create a useable array structure. However, the use of replication to achieve redundancy does make the structure very silicon-inefficient.

A more efficient approach to achieving high cell yields is achieved in for example, the 'n out of m' approach used in the HYETI architecture. The processor is implemented as a bit-slice structure and hence permits a small number of interchangeable spare-bit slices to improve cell yield greatly, with only a small increase in area. Although this approach requires more complex test and configuration circuitry.

2.6.4 Comparison of interconnection strategies

It is difficult to compare the cost-effectiveness of different interconnection strategies due to the wide variety in technologies and implementation methods, combined with the differing forms in which results are published. However, Jones and Lea; [64] report on some WSI architectures for which comparable figures are available. The results of this comparison are given in Table 2-3.

The exceptionally low area overhead of the Aubusson/Catt and Manning approaches (where a grid of nearest-neighbour connected cells form the desired structure), is offset by a poor cell utilisation.

Architecture	Area Overhead	Cell Harvest at 70% yield
Aubusson/Catt	1%	25-45%
CHiP	66%	99%
Manning	1%	42% (spiral)
RPA 16 x 16 array	25% 4 spare rows	78%
RVLSI FFT	30%	52%

Table 2-3. Area overheads and cell harvest.

The laser-zapping approach of RVLSI requires a large area overhead for the 'manhattan mesh' of wires required for connectivity. The cell utilisation is determined by the requirement for an exact number of cells (as the wafer is a 16-point FFT processor). For other applications, a larger proportion of cells could be used. Cell harvest and communication optimisation for this approach is a current research topic. While this approach seems promising, the cost of laser 'zapping' equipment and the individual processing of each wafer may make it cost-ineffective.

The hierarchical approach of the CHiP wafer results in a very high cell utilisation. However, this is achieved at the cost of a large area overhead (approximately 66% of the total wafer area is unproductive as each cell is triplicated for reliability).

In contrast, the figures from the RPA investigation strikingly

demonstrate the cost-effectiveness of an 'n out of m' approach, with only 25% area overhead, 78% of 16 x 16 arrays can be realised.

These results would seem to indicate therefore, that in terms of fault-tolerance, hierarchical and 'n out of m' schemes, in general, appear to be the more silicon-efficient interconnection strategies.

2.7 Selection of representative structure

There are a number of architectural components that could be considered as representative of WSI fine-grain parallel processing architectures (eg. a systolic cell or a bit-serial processing element). However, it is interesting to note that two of the DLM architectures (viz. ALAP and WSI-DLM) which have quite different application areas, make extensive use of associative memory. Indeed, as the following chapter demonstrates, an associative memory is a flexible parallel processing structure capable of supporting a wide range of computational tasks. Furthermore, while the design of a systolic cell or bit-serial PE may change significantly depending on the application, an associative memory remains essentially the same structure (viz. a data store plus comparison logic) across different applications. It is for these reasons then, that associative memory has been selected as a representative structure for WSI investigations.

2.8 Conclusions

This chapter has surveyed a wide range of WSI architectures. While WSI may be used for application-specific circuitry, it also offers much potential for the implementation of large parallel processing

systems. However, no large parallel processing system has yet been built in WSI using modern integrated circuit technology. Consequently, many of the electrical and physical design issues of WSI have not been addressed.

An important objective of this chapter has been to analyse WSI architectures to identify a set of WSI-level design requirements for the research vehicle. These requirements may be summarised as follows.

Standard technology should be used. This provides the lowest cost option. A consequence of using standard technology is that 'soft' electronic configuration must provide the fault-tolerance. To reduce test costs then, the research vehicle must also incorporate self-test logic. Also to provide low fabrication costs, the design should use as few reticles as possible and require only periphery pinout.

The research vehicle should implement silicon-efficient fault-tolerance. From the architectures reviewed this implies that a hierarchical interconnection strategy needs to be used. Furthermore, the fault-tolerance should use 'n out of m' techniques.

The architectures reviewed comprise 3 main categories: systolic arrays, processor arrays and DLM's. While there are many architectural components that could be considered as representative WSI structures. Associative memories have been selected as the structure for investigations. In relation to this, the following chapter demonstrates how an associative memory can be used to flexibly support a wide range of applications.

CHAPTER 3 ASSOCIATIVE PROCESSING

3.1 Objectives

The objective of this chapter is to demonstrate how an associative memory may be used to flexibly support a wide range of both numerical and non-numerical applications.

3.2 Overview

It is a common requirement in information processing for a set of one or more data elements associated with a common key (eg. the personnel records of all computer programmers in a company) to be selected for subsequent processing. Unlike conventional processing, where data elements are accessed sequentially by their location in the data structure, associative processing (where data is content-addressed, rather than location-addressed) is a natural form of processing offering benefits of ease of expression (cf. conventional processing, where elaborate data structures need to be set up and maintained to emulate associative access), storage capacity (reduction in data structure complexity) and speed of execution (parallelism of associative processing).

Associative processing can be realised by an associative processor (Figure 3-1.), which comprises a large associative memory supported by control logic for the flexible access and manipulation of the associative memory data.

Conceptually, in operation an associative processor transmits a search vector (with maskable data fields) to the associative memory.

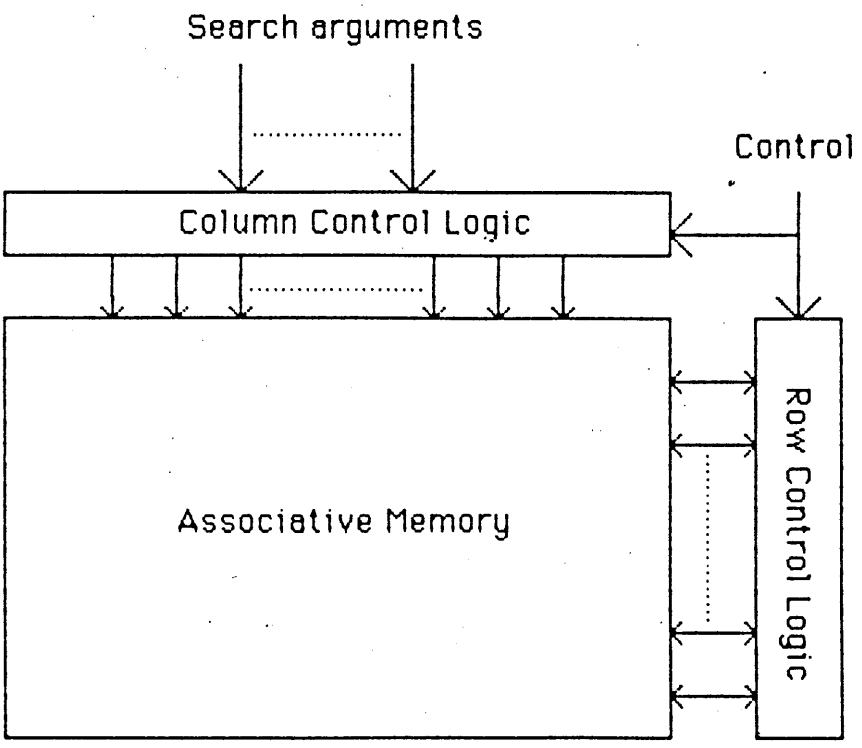


Figure 3-1. Associative processor schematic.

This search vector is simultaneously compared in each word-row of the associative memory against the data contents of that word-row and a vector generated corresponding to the set of word-rows that matched the search patterns. This match vector is transferred to the word-row control logic and corresponds to the set of data elements (0 or more) selected for subsequent processing. Typically, this processing is accomplished by transferring data, in parallel, from the associative memory to the word-row control logic, manipulating the data and writing it back to the associative memory. As the following section shows, this basic associative process can be used in a wide range of applications.

3.3 Applications

The flexibility and parallelism of associative processing enables it to be applied to a surprisingly large range of applications.

3.3.1 Set processing

Associative processing is effectively a form of set processing, each word of associative memory can be considered as an element of a set. using the pattern matching capabilities of the associative memory, an un-ordered sub-set of unknown cardinality can be selected for subsequent set processing operations (eg. union, intersection, inclusion).

3.3.2 String processing

Text string processing is perhaps the most obvious form of non-numeric application for associative processing. For example, to reflect a departmental name change in personnel records, all

occurences of 'Electrical engineering' need to be replaced with 'Electrical and Electronic engineering'. In an associative processor, the string 'Electrical Engineering' would be searched for, then the string 'Electrical and Electronic Engineering' would be written (in parallel) to all word-rows that matched the previous search argument.

3.3.3 Array processing

Associative processing can also support vector processing. Each word-row (with non-selected data fields in the word row masked) can be considered to be an element of a linear vector. Consider the following examples.

Assuming scalar value 'x', and the vector 'B' (where B has been previously selected on the basis of a search pattern presented to the associative memory), then in pseudo-pascal

```
forall i: (selection criteria) do
  A[i] := x + B[i];
```

thus each word-row that matched the search argument (elements in the vector that met the search criteria) would be incremented by the scalar quantity x. The addition could be implemented in many ways, but is usually performed by attaching a simple arithmetic unit to each associative memory word-row, transferring data from the word-rows (in parallel) into the arithmetic units, computing the results and writing them back (in parallel) into the appropriate data field.

Vector-vector operations can similarly be formed. As in the previous example, if 'B' and 'C' are the vectors (which correspond to two fields of selected data records), again in pseudo-pascal

```
forall i : (selection criteria) do
  A[i] := B[i] + C[i];
```

Here, for example, two fields of the same data record are added together.

Thus, associative processing using the basic search for a set of 'active' word-rows (viz. those words in the associative memory that match the search arguments) and manipulation of memory contents (in parallel) operation, can flexibly support scalar-vector and vector-vector operations.

3.3.4 Signal processing

The vector processing outlined above can be widely used in a range of signal processing operations. Examples of these include,

(1) Correlation, where a sampled input signal is matched against a specified reference waveform.

(2) Convolution, where a sequence of input signals is convolved with a set of weights that correspond to a specified filter block.

3.3.5 Image processing

The fundamental scalar-vector and vector-vector operations of supported by associative processing are of course, widely applicable to a range of image processing operations. Examples of this include,

(1) Histogramming, where a count is made of the number of pixels having a specified brightness value to support contrast

value.

3.3.6 Relational data processing

A schematic description of a relation is shown in Figure 3-2. A relation comprises a set of tuples (viz. entries in the relation table) each tuple is composed of a number of attributes, (cf. data fields in a record). The relational data model enables a set of data elements associated with a common attribute to be referenced by virtue of the value of that attribute (ie. independent of its location in the relation).

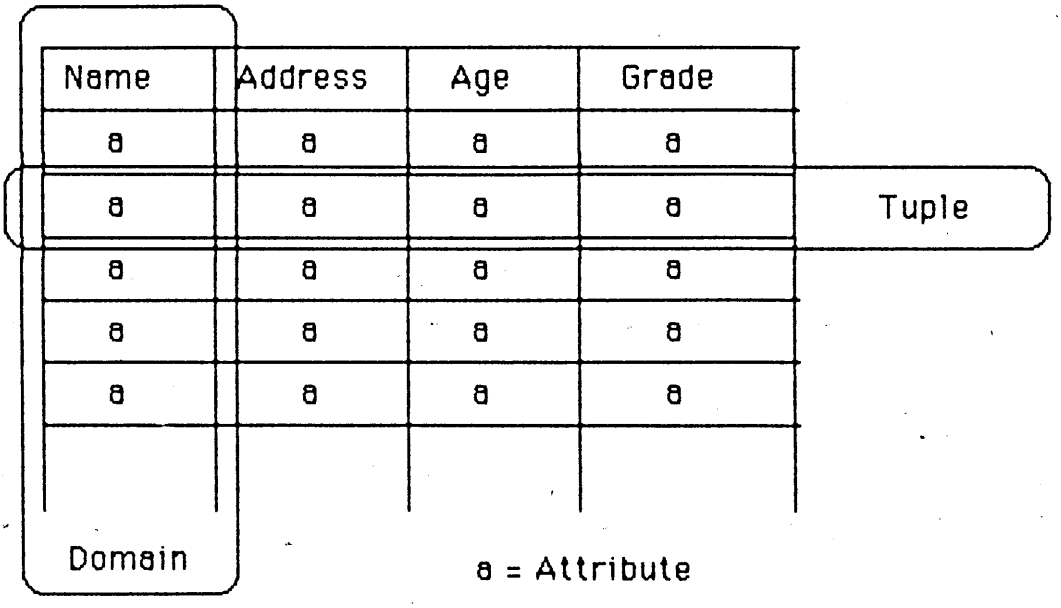


Figure 3-2. relational data model.

Relational data is processed through a relational algebra which involves the creation of a new relation from either 1 or 2 existing relations. In essence such relational data processing is similar to associative set processing, where a sub-set of unknown cardinality (viz. the tuples in a relation who have matching attribute-domains) can be selected for subsequent set processing operations (eg. intersection, union).

3.3.7 Fifth generation language support

The capability of associative processing to support rapid and flexible pattern matching makes it suitable for the support of fifth generation languages such as PROLOG. Which, as the following example shows, involves interrogating a database of facts and rules.

```
Red_wine (Claret).
Red_wine (Chianti).
White_wine (Muscadet).
Meat (Beef).
Meat (Pork).
Fish (Sole).
Drink_with (X,Y):- Red_wine (X), Meat (Y).
Drink_with (X,Y):- White_wine (X), Fish (Y).
```

Typical PROLOG questions might be,

```
?- Drink_with (Muscadet, Sole).     answer = yes
?- Drink_with (Muscadet, Beef).     answer = no
?- Drink_with (X, Pork).            answer = Claret, Chianti.
```

Clearly, the basic feature of these operations involve pattern matching the query (with the 'X' and 'Y' fields masked out) against a

set of data entries to identify the matching facts and rules.

3.4 Associative processor implementation

In all the above examples, it has been assumed that the associative memory is wide enough to accomodate the largest record size and large enough to hold all records simultaneously. In practice, the wide diversity of applications means that to achieve this objective, the associative memory would have to be very large. Moreover, except for special purpose applications, most of the associative memory would be unused most of the time. As a consequence of this, most practical associative processors support multi-word allocation for data records and virtual associative memory management.

3.5 Associative string processing

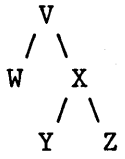
Associative string processing is a particularly flexible form of multi-word allocation. This arises from the fact that all data structures can be mapped into a string (cf. the storage of structured data in a disk file) allied to the content-addressing capability of the associative memory supporting flexible access to this data.

The members of a set, text strings and linear arrays are essentially one-dimensional data structures and hence all map naturally into a string format. More complex data structures can also be supported by the associative string. For example, n-dimensional arrays can be represented by concatenating the linear arrays associated with each dimension and separating groups of data items (X) with reference markers (M) eg. for a 3 x 3 array

$$M\ X_{1,1}\ X_{2,1}\ X_{3,1}\ M\ X_{1,2}\ X_{2,2}\ X_{3,2}\ M\ X_{1,3}\ X_{2,3}\ X_{3,3}\ M$$

Relational data models can be concatenated in the same way into the associative string format.

Tree structures also map straightforwardly into the associative string format. For example,



maps to (V W (X (Y Z)))

Any linked list may also be represented as a string by using the following data storage format.

in-tags	data	out-tags

The data structure could be traversed by simple associative searches using the tags as search patterns. For example, the tree structure given above may be stored as

v V w x w W 0 0 x X y z y Y 0 0 z Z 0 0

since this is for a binary tree, every data element has 1 in-tag and 2 out-tags.

To summarise: associative string processing representation and manipulation is as equally universal as list processing. However, associative data access does not require the complex traversal of pointer networks to identify the physical location of data items that list processing requires, but uses the parallel search capability of the associative memory to access and manipulate data.

3.6 Conclusions

This chapter has shown associative memory to be a general-purpose flexible paralllel processing architectural component. In particular associative string processing is argued to be a cost-effective implementation of generalised associative processing.

The next chapter introduces a particular implementation of an associative string processor, the WSI-ASP (WASP) device, that has as its core a large associative memory. The WASP device is evaluated against the WSI-level design requirements developed in the previous chapter to determine its suitability as a research vehicle.

CHAPTER 4. WASP - A WSI ASSOCIATIVE STRING PROCESSOR

4.1 Objectives

This chapter has the following objectives,

(1) To introduce the WASP architecture, focussing on the proposed WASP image processing device.

(2) To report the results of a study comparing WASP and its VLSI equivalent, in order to show the potential of the WASP device and to further clarify the advantages gained from using WSI.

(3) To place the WASP device within the context of other WSI parallel processors.

(4) Evaluate the WASP device against the WSI-level design requirements exposed in Chapter 2 and hence to appraise its suitability as a research vehicle for WSI investigations.

4.2 WASP

WASP (WSI-ASP) [65,66] is a wafer-scale implementation of an associative string processor (ASP). An ASP is a variant of the distributed logic memory (DLM) architecture originally proposed by Lee [67], and since then developed by many researchers including Savitt, Love and Troop [68] (who first used the term ASP, but for association storing processor), Sturman [69], Beaven and Lewin [70], Finilla and Love [19], Lea and Wright, [71], Lea and Sreetharan [60] and Mukhophadyay [72].

4.2.1 Associative string processor

An associative string processor is a parallel processing computational structure, comprising a linear string of identical associative processing elements (APEs) that operate in parallel to achieve high-speed cost-effective structured data processing (Figure 4-1.). Each APE is connected to an inter-APE communication network, that runs in parallel with the APE string. All APEs are connected to common data, activity and control busses and share a single feedback line (MR). These busses and the MR line are supported by an external string controller, which also maintains the left (LAP) and right (RAP) activity ports of the inter-APE communication network.

Each APE comprises a data register, an activity register and a comparator (implemented through an associative memory) together with logic for local processing and communication with other APEs in the string.

4.2.2 Associative string processor operation

In operation, the associative string processor executes two classes of instructions, namely

(1) APE activation functions: during which all APEs simultaneously compare the contents of their registers with the values on the data and activity busses. Matching APEs either have their activity registers updated ('activation') or start inter-APE communications to indirectly activate other APEs, according to the instructions broadcast on the control bus.

(2) Active APE procedures, during which all active APEs simultaneously execute local processing operations. In the

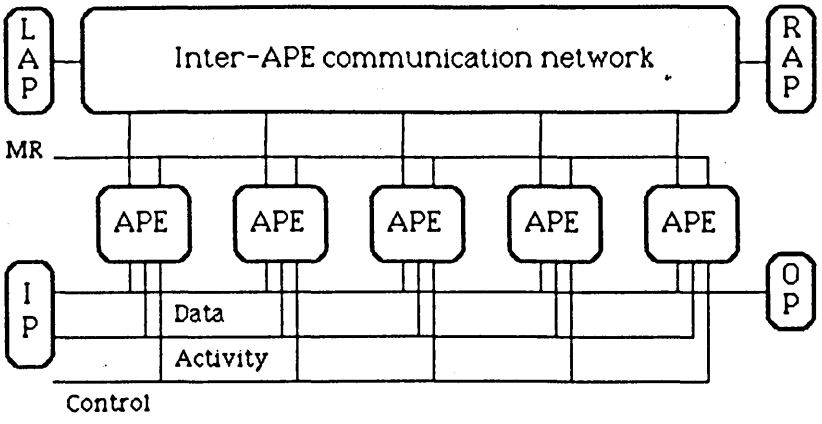


Figure 4-1. Associative string processor.

simplest APE implementations, this processing is restricted to changing the states of the data and activity registers (writing) or transferring the contents of these registers to the data bus (reading). However, in more sophisticated APEs, extra hardware to support logic operations such as addition and subtraction may be implemented.

4.2.3 Associative string processor hardware

Lea, [73] has shown that a segment of an associative string processor can be cost-effectively implemented with an associative parallel processor (APP). As shown in Figure 4-2., an APP comprises a large associative memory array (AMA) of content-addressable memory cells (CAMs) together with microprogrammably controllable bit (BCL) and word (WCL) control logic. Relating Figures 4-1. and 4-2., it can be seen that each APE is implemented as one word slice of the AMA (data and activity registers), with the processing logic implemented in the WCL. In addition, the WCL also incorporates the inter-APE communication network. The BCL broadcasts data and activity information to the APEs and the micro-order generation logic (MOGL) distributes micro-orders to the BCL and WCL to simplify the control of the APP. The IOI interface connects the APP to the outside world.

4.2.4 WASP philosophy

The philosophy of the WASP device is to achieve WSI implementation using standard VLSI technology, in order to benefit from the investment already made in IC manufacturing technology. Accordingly, this implies that WASP must use soft electronic configuration to achieve fault-tolerance (eg. no laser zapping or E-beam programming).

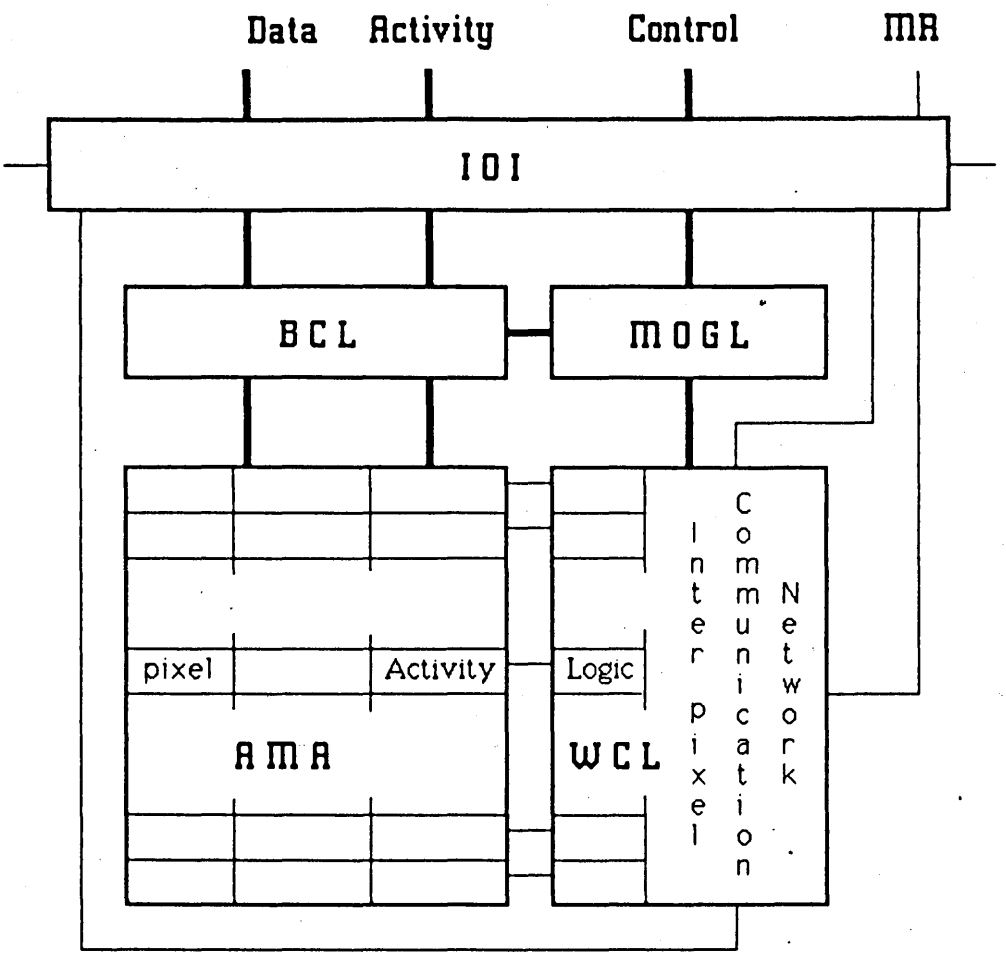


Figure 4-2. Associative parallel processor.

A 2-micron, 2-layer metal, p-well bulk-CMOS process is currently used in the design of WASP devices.

4.3 A wafer-scale image processor

The furthest developed WASP variant to date, is a version targetted at the support of image processing applications (WASP IP) and consequently, has been selected as the particular WASP device to evaluate.

To provide necessary background information, the following section introduces image and patch processing modules.

4.3.1 Image and patch processing modules

A typical image processing module (IPM) to be applied as an intelligent peripheral to a host computer is shown in Figure 4-3., and comprises four main elements

(1) Image store (IS), providing buffer storage for a number of video frames.

(2) Patch Processing Module (PPM), supporting high-speed parallel processing of sub-image patches transferred from a selected video frame.

(3) Micro-controller (MC), supporting pixel I/O, patch selection and transfer, together with sequential processing of intermediate results.

(4) Input-Output Interface (IOI), for the support of independent input (eg. TV camera) and output (eg. display monitor), together with a communications link to the host computer.

For real-time image processing, the requested sequence of patch processes must be executed within one image time frame (viz. 40ms for interlaced-raster TV and 50Hz mains frequency), a processing requirement in excess of 1 Giga adds-per-second. A PPM that can meet this processing requirement based on SCAPE chips is shown in Figure 4-4. A chain of SCAPE chips is linked to a patch buffer (PB) and controlled by a SCAPE chain controller (SCC). External control and data is handled by the Input-Output Interface (IOI).

4.4 WASP image processing device

The image processing WASP variant aims to integrate an 8192-pixel, real-time patch processing module, based on the ASP architecture, on a single, undiced, 4 inch silicon slice. The patch is reconfigurable into any 2048/n row by 4n column patch, where n is a non-zero positive integer and not greater than 2048 (eg. 1024 x 8, 512 x 16, 128 x 64).

Relating the patch processing module (Figure 4-4.) to the WASP floorplan (Figure 4-5.), it can be seen that the WI (wafer interface) modules implement the IOI. The CC (control and communication) modules provide the SCAPE chain controller functions and the RAM modules implement the patch buffers. Finally, the APP modules replace the SCAPE chips.

In operation, a 2-dimensional patch is mapped into the associative string. Thus a 512 x 512 image frame could be processed by 1 'sweep' of a 512 x 16 image patch across the frame. The RAM modules support multiple patch storage and patch transfers between the RAM buffers. External frame store access can be overlapped with patch processing within the APP modules to achieve high-speed patch processing.

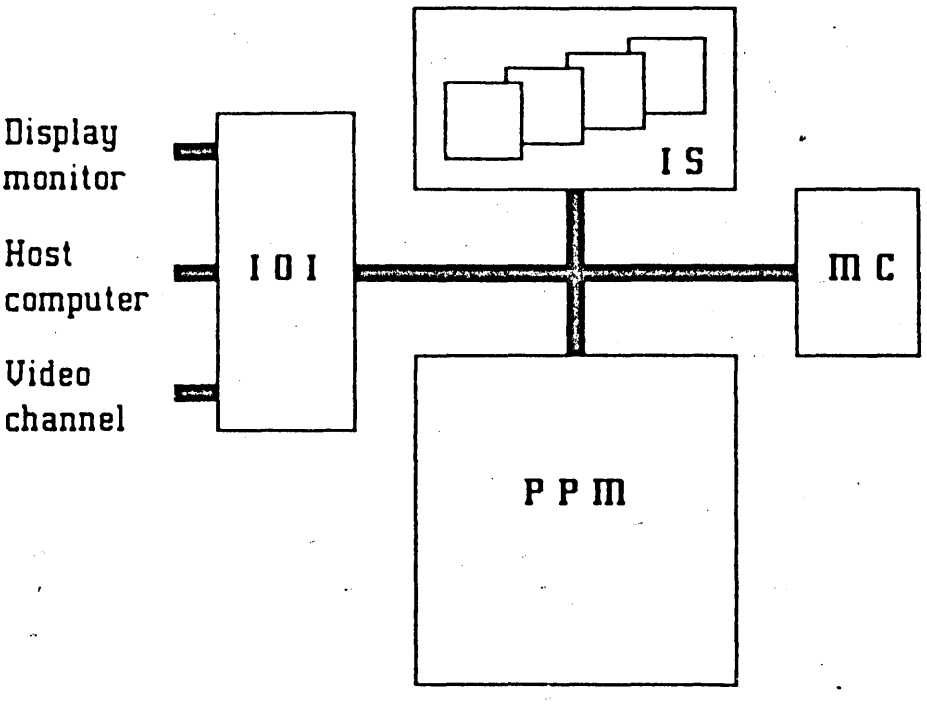


Figure 4-3. Image processing module.

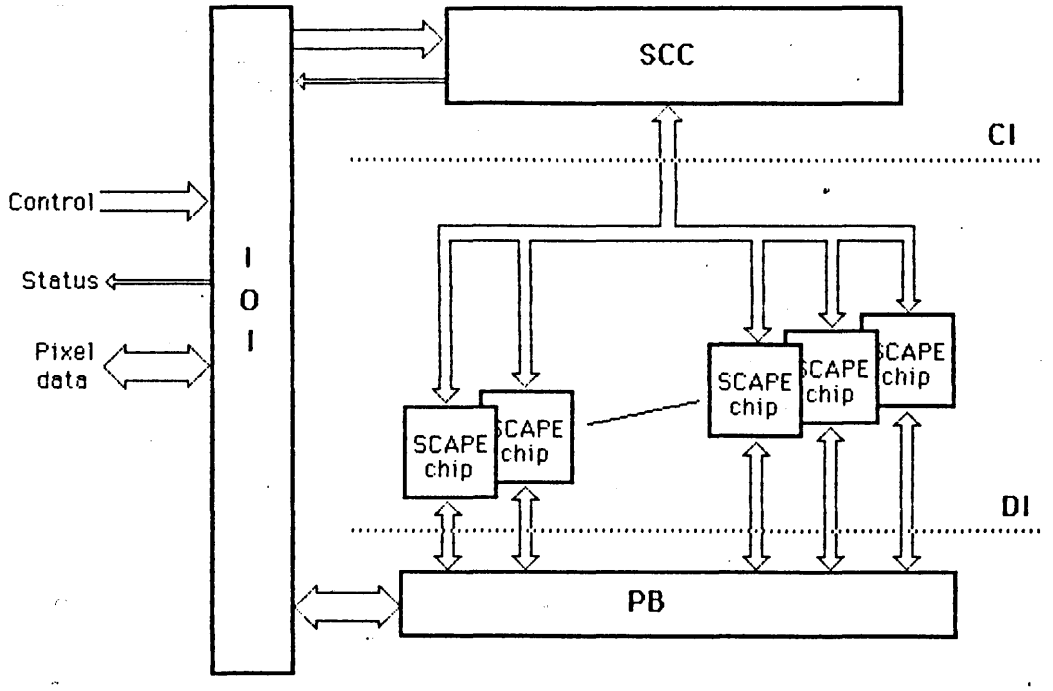


Figure 4-4. Patch processing module.

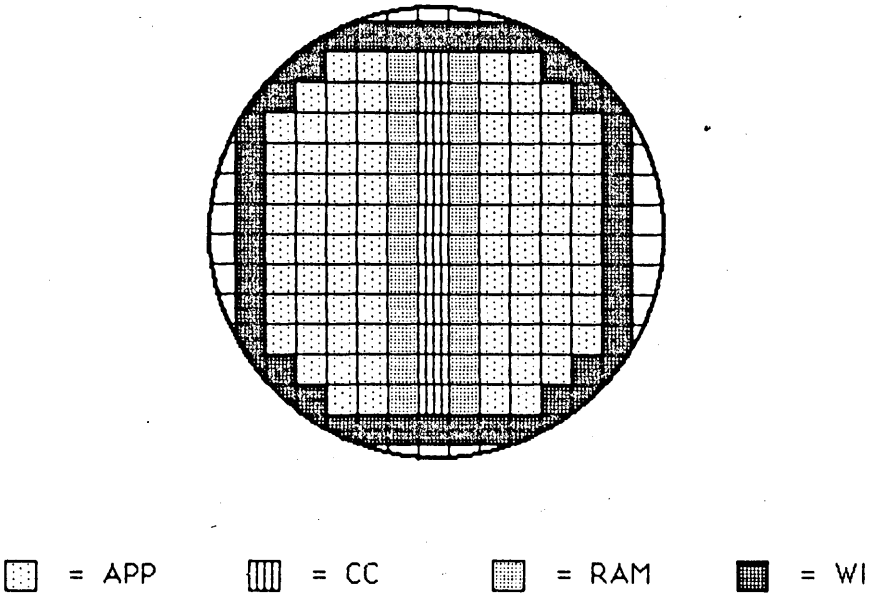


Figure 4-5. WASP floorplan.

4.4.1 Wafer interface modules

The WI modules provide off-wafer connections. The WI modules support 8 (8-bit) pixel channels together with control and power connections. Total WASP pinout is expected to be 148, comprising

(1) 64 pixel IO pins.

(2) 36 control pins.

(3) 48 power pins.

4.4.2 Control and communication modules

These modules provide the broadcasting of instructions to the APP modules together with supporting pixel transfers both between the APP and the RAM module and also from the RAM modules to the 8 external pixel channels. In addition, the CC modules support long-distance APE communications (viz. between APP modules that are controlled by different CC modules).

4.4.3 RAM modules

These modules provide the storage of pixels and allow pixel data to be exchanged with the APP modules and the 8 external pixel channels.

4.4.4 Associative parallel processor modules

The APP module (Figure 4-2.) is a fault-tolerant (viz. the ability to bypass small groups of APEs - which is implemented through the inter-APE communication network), and smaller (viz. fewer APEs per APP to achieve a intrinsically high yield) version of the SCAPE chip

architecture. Each pixel is allocated a 32-bit data word and 5 activity bits, which forms one word of the associative memory array (AMA). Each WCL slice comprises a 1-bit ALU for each pixel and an inter-pixel (APE) communication network. The BCL supports data masking and index control logic for bit-serial arithmetic. The MOGL broadcasts sequences of micro-orders to execute instructions given by the CC modules.

4.5 WASP interconnection strategy

WASP uses a hierarchically organised tree-structured interconnection strategy The root of the tree is the WI, the branches are the CC blocks, and the twigs RAM blocks, with the leaves being the APP blocks. Faulty groups of APEs, APP modules, or even complete branches are bypassed by re-routing their LAP and RAP activity ports. As Figure 4-6. shows, by connecting the LAP and RAP ports together appropriately, a long associative string can be configured. Since all APEs can be used, regardless of their location, the WASP fault-tolerance strategy is effectively an 'n from m' strategy.

A major advantage of the tree structure is that it allows the fault-tolerance to be partioned according to the 'criticality' of the module in the tree structure, with the modules closest to the root of the tree receiving most fault-tolerance. However, there are only a few critical modules (eg. 6 WI blocks) and hence applying a large amount of fault-tolerance to these modules, significantly improves cell harvest, but only requires a modest proportion of the wafer area to do so.

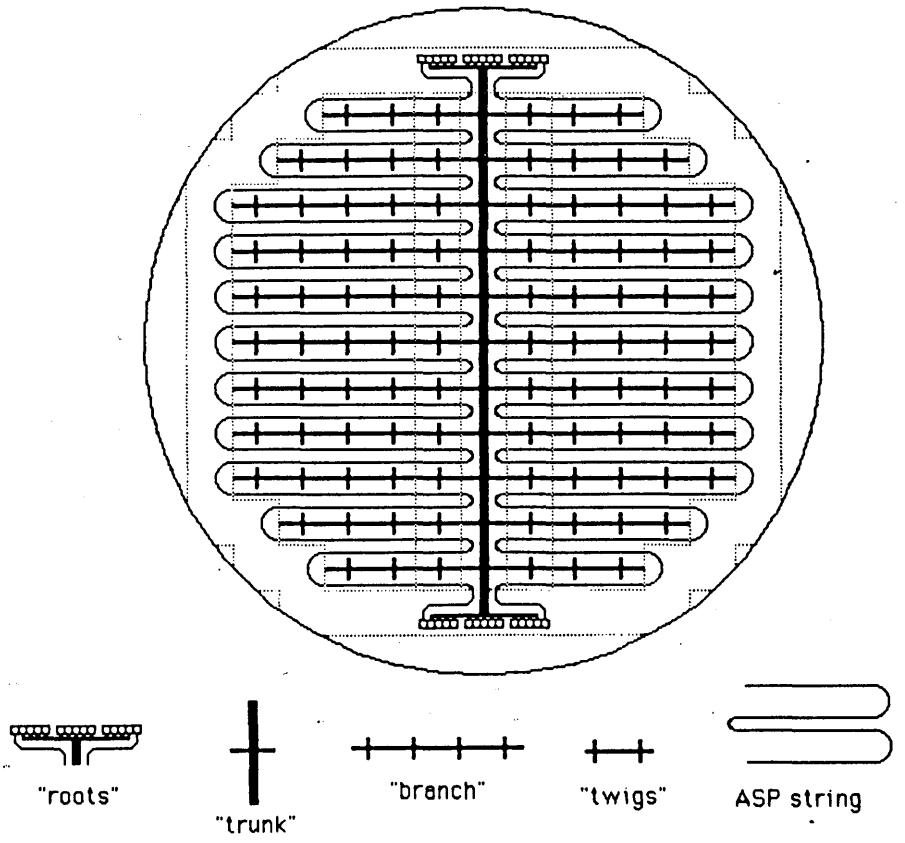


Figure 4-6. WASP associative string.

The tree interconnection strategy offers many advantages for WASP specifically,

(1) Each 'twig' APP is a self-contained string segment, receiving data and instructions from its CC block. Larger string segments can be formed by activating the required set of LAP and RAP links and CC blocks. Each CC block can contain identical, or different, control microprograms for SIMSIMD or MIMSIMD operation.

(2) High-speed operation is supported through using the 'logarithmic' communication delay of a tree structure to provide high-speed inter-APE communications.

(3) The tree is fault-tolerant, since any failures affect only those sub-trees 'below' the point of failure. Moreover, replication of WI, CC and RAM block logic to improve fault-tolerance is easily accommodated and relatively inexpensive in terms of total silicon area.

(4) The tree is composed of only 4 structure block types (viz. CC, APP RAM and WI blocks), reducing mask making costs and allowing simple 'step-and-repeat' fabrication techniques to be used.

A more detailed treatment by Jones and Lea of the merits of the tree interconnection strategy can be found in reference [64].

In keeping with the use of standard IC technology for WASP fabrication, WASP uses soft electronic switches to implement fault-tolerance. Indeed, this permits 'graceful degradation', allowing in-service failures to be survived by configuring around the failure site.

WASP is targetted to integrate 196 x 64-APE APP modules, giving 12544 APEs in total. Consequently, to achieve an 8192-APE wafer,

$$\frac{8192}{12544} = 65.3\%$$

of the APEs must be harvested. However, due to the variation in defect density from wafer to wafer, some of the WASP devices will have less than 8192 APEs. These devices will be used as WASP 4000 (viz. a wafer comprising 4096 working APEs) devices. More heavily flawed wafers will be used as WASP 2000 (2048 working APEs) and 1000 (1024 working APEs) wafers. Furthermore, the WASP device is not restricted to using APES in multiples of 1024. If the application can make use of them, all working and accessible APEs may be utilised. Thus, it is anticipated that efficient use will be made of almost all wafers produced.

In addition, the WI, CC, RAM and APP modules may be composed in alternative formations to create ULSI (viz. sub-wafer) ASP devices.

4.6 WASP testing

It is anticipated that WASP will make significant use of self-testing in order to reduce total test time and test costs. Indeed, the parallel operation of the APPs and the tree interconnection strategy facilitates this. Each APP module can test itself in parallel, with amortisation of test logic among many APPs being achieved through placing test logic at the controlling CC level. On power up, an ASP string is 'grown' starting from the root nodes (viz. the WI blocks) continuing through successive levels (viz. the CC and RAM blocks) up to the final incorporation of functional APP blocks. Furthermore,

WASP offers the potential to combine the three levels of system testing (viz. wafer probing, chip test and overall system test) into a single WASP test.

4.7 WASP electrical design issues

Electrical design issues in WSI, concern power, clock and signal distribution on the wafer.

4.7.1 WASP power distribution strategy

A WSI power distribution strategy must maintain voltage integrity across the complete wafer under all operational conditions. Moreover, it must be fault-tolerant since a failure in the power distribution network could affect the complete wafer.

In [74] Warren, Abdelrazik, McKirdy and Lea, demonstrate that an independent rails network appears to be attractive to WSI implementation for the following reasons.

(1) The structure is fault-tolerant since power distribution is divided amongst several networks and a failure in one network only affects a portion of the wafer

(2) Only one layer of metal is required, allowing other layers to be freely utilised for control and data distribution

(3) It is amenable to low cost fabrication, using conventional 'step-and-repeat' technology as the physical positioning of the rails in each reticle is identical.

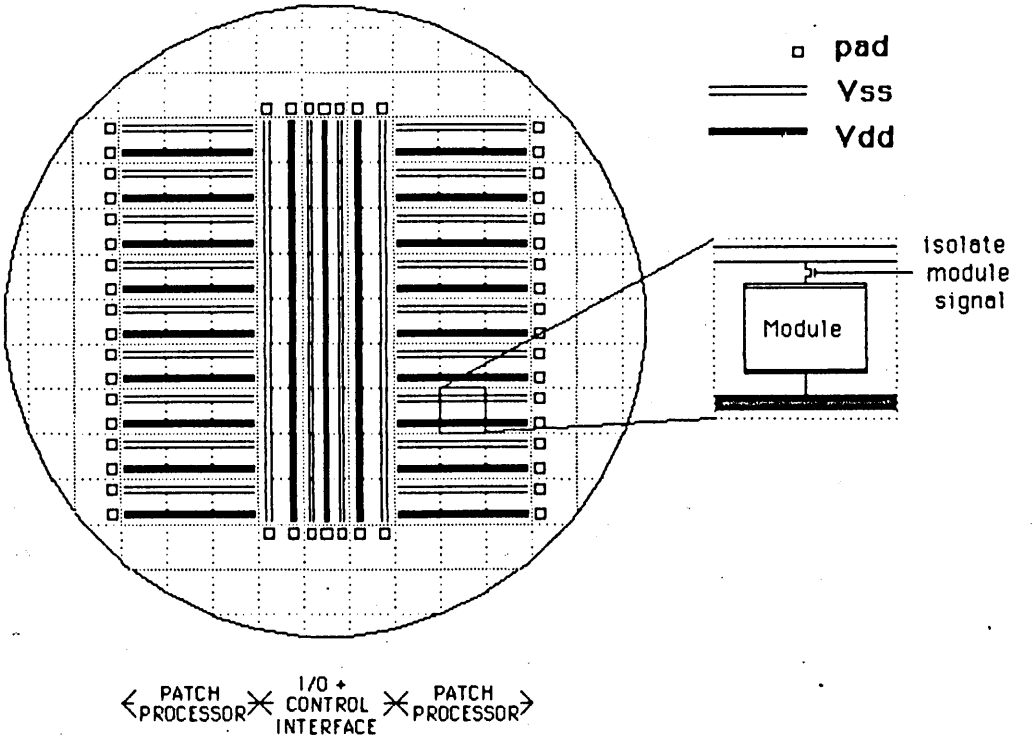


Figure 4-7. WASP power distribution strategy.

Consequently, an independent rails power distribution strategy has been selected for the WASP device that closely matches the WASP tree interconnection strategy (Figure 4-7.). In addition to the multiple power networks, soft 'electronic' switches are used to disconnect modules (eg. ASP sub-strings) from the power network should shorts occur internally to the module. This combination of multiple power networks and isolating 'leaky' modules can provide a very highly fault-tolerant power distribution strategy. At the time of writing, one test chip is in processing and another is being designed, to provide quantitative information as to the effectiveness of the WASP power distribution strategy.

4.7.2 WASP clock distribution strategy

The long RC transmission lines in WSI significantly limits the clock speed in WSI. Moreover, the systems designers requirement for a slow-speed wafer interface, further complicates the issue of clock distribution.

The WASP clock distribution scheme parallels the WASP tree interconnection strategy and makes extensive use of on-wafer clock multiplication. All on/off wafer communication, together with communication using the long trunk bus running through the CC modules, occurs at 10 MHz. The shorter CC/RAM-APP busses operate at 20 MHz. Intra-APP communication occurs at 40 MHz.

This approach offers the advantages of a simple slow-speed system interface together with high-speed processing, while still maintaining clock integrity across the wafer.

4.7.3 WASP signal distribution strategy

The tree interconnection strategy also allows wide-bandwidth signal transmission. Since the potential bandwidth increases greatly further down the tree. In addition, there are wide signal busses in the RAM-APP branches allowing even higher data communication rates to be supported. The busses are also fault-tolerant, using an 'n out of m' approach and self-configuring signal distributors at either end of the bus to create a working bus [75].

4.8 WASP physical design issues

The problems of packaging a device two orders of magnitude larger than current VLSI chips are significant. The factors related to WSI packaging under the control of the wafer architect are pinout and power.

It has already been noted that the pinout of ASP architectures is independent of string length and a consequence of this, is that WASP only requires 148 pins. Furthermore, the ASP architecture and the tree fault-tolerance strategy allows unused wafer elements to be powered down, thus reducing wafer power. Consequently, the combination of these two factors does simplify the WASP packaging problem.

4.9 Comparative appraisal

In reference [76], McKirdy, Abdelrazik, Bolouri, Hedge, Jones, Warren and Lea compare the proposed WASP device against the proposed VLSI (SCAPE) patch processing module. The results of which are summarised here

Speed: WASP offers 64-128 speed improvement over VLSI

Cost:

Silicon-efficiency: to realise a PPM in WSI requires 2 wafers (assuming 50% wafer yield), in VLSI it requires 3.7 wafers.

Materials:	WASP	VLSI
Packages	1	128
Area (Si) (sq in)	8	5.5
Area (foot print) (sq in)	9.43	79.4
Package density (area(Si)/area(FP))	84.6	6.9
Pin-count	148	4392

Assembly: greatly reduced cost due to 30-fold decrease in system pin-count.

Testing: 3 levels of VLSI testing compressed into one WASP test sequence.

Power: 3-4 times lower system power in WASP.

Size and weight: order of magnitude improvement.

Reliability: 30-fold decrease in WSI system connections, together with the possibility of utilising spare WASP APE's to implement 'graceful degradation'.

This data clearly demonstrates the significant improvement in cost-effectiveness and performance achievable if the proposed WASP device is feasible.

4.10 Comparison and evaluation

The previous sections have discussed the WASP philosophy and architecture and shown the improvement in cost-effectiveness available from the use of WSI for ASP implementation. This section places the WASP device within the perspective of other WSI parallel processor architectures and evaluates WASP against the WSI-level design requirements exposed in Chapter 2 to determine the suitability of WASP as a research vehicle for WSI investigations.

Table 4-1., compares the architecture, fault-tolerance technology and interconnection strategy of the WASP device with the range of WSI processor architectures reviewed in Table 2-1. of Chapter 2.

WASP, like COBWEB, Aubusson/Catt and Manning's approaches, uses conventional fabrication technology and thus can achieve cost advantages from the use of currently available IC fabrication lines.

Curiously, while many architectures (eg. RPA, CHiP) use a hierarchic (usually a 2-level hierarchy) scheme for fault-tolerance, WASP appears to be the only system that incorporates a multi-level tree interconnection strategy (viz. WI-CC-RAM-APP-APE) both for fault-tolerance and for communication.

In terms of fault-tolerance, Table 4-2., reports some results from a study by Jones and Lea [64] that compares a prototype WASP device that has $1cm^2$ CC blocks and no RAM modules with the fault-tolerance figures in Table 2-3.

Name	Architecture	Fault-tolerance technology	Interconnection strategy
ALAP	DLM	discretionary	string
Aubusson/ Catt	DLM	standard tech	string
CHiP	processor array	standard tech	hierarchic array
COBWEB	DLM	standard tech	string
HYETI	processor array	laser/E-beam	array
Manning	variable	standard tech	variable
Moore	systolic array	standard tech	array
RPA	processor array	split bondpads	hierarchic array
RVLSI	special-purpose	laser zapping	customised manhattan mesh
Hedlund's WASP	systolic array	standard tech/ laser/E-beam	hierarchic array
WINNER	variable	standard tech	array
WSI-DLM	DLM	standard tech	string
Brunel WASP	DLM	standard tech	tree

Table 4-1. Comparison of WASP and other WSI architectures.

From Table 4-2., it can be seen that in fault-tolerance terms WASP benefits significantly from its hierarchical tree interconnection strategy and 'n out of m' redundancy scheme. It has a modest area overhead. Indeed, this area overhead figure is only this high as the CC blocks are counted as unproductive logic. If the CC blocks are counted as processing logic (since they contain the control

microprogram and scratchpad RAM) then the area overhead is significantly lower. The expected WASP harvest is only exceeded by the CHiP approach. However, the CHiP approach uses 66% of the total wafer area (as compared with 14% for WASP) for intra-module redundancy to reach this figure.

Architecture	Area Overhead	Cell Harvest at 70% yield
Aubusson/Catt	1%	25-45%
CHiP	66%	99%
Manning	1%	42% (spiral)
RPA 16 x 16 array	25% 4 spare rows	78%
RVLSI FFT	30%	52%
WASP	14%	97%

Table 4-2. Area overheads and cell harvest.

While the WASP figures are for a slightly different variant, they do demonstrate the efficiency of the WASP tree interconnection strategy. Furthermore, the estimated 97% APE harvest appears encouraging for the image processing WASP variant where a 65.3% harvest is needed.

In terms of testing and electrical and physical design issues, there are few grounds for comparison with other WSI architectures. However, from the description of the WASP IP device it appears prima facie, that realistic engineering approaches have been adopted.

4.11 Conclusions

The WASP device has been introduced and a detailed description of the

WASP IP philosophy, architecture and operation has been given. A comparison of WASP against the equivalent VLSI implementation, has shown that WSI offers many potential benefits for ASP integration. This comparison suggests, that WSI offers the potential to improve the cost-effectiveness of ASP-based image processing modules by at least an order of magnitude.

In terms of the WSI-level design requirements exposed in Chapter 2, the use of standard fabrication technology suggests that WASP offers the potential for low-cost implementation. A comparison of WASP against other WSI architectures has suggested that its hierarchic tree interconnection strategy appears to be an exceptionally efficient fault-tolerance strategy. The WASP test scheme appears to offer the potential for cost-effective self-test techniques to be applied. Furthermore, at this initial level, cost-effective techniques appear to be proposed for WASP that, prima facie, satisfy the electrical and physical design constraints of WSI.

From this broad perspective then, it does seem reasonable to hypothesise that WASP appears to meet the WSI-level design requirements and that WSI implementation could significantly improve ASP cost-effectiveness when compared to a VLSI implementation of the same system. The proposed WASP device then, is judged to be a suitable research vehicle for WSI investigations.

Having identified and evaluated a suitable research vehicle, this thesis now aims to investigate the potential of WSI for implementing low-cost fine-grain parallel processors from an 'in-depth' perspective, by investigating the implementation of a specific aspect of WASP, the large associative memory, through a set of

investigations to determine whether a suitable associative memory for WASP can be cost-effectively engineered to meet the technological constraints of WSI.

The next chapter describes the investigation strategy taken to investigate the implementation of the WASP associative memory.

CHAPTER 5. INVESTIGATION STRATEGY

5.1 Objectives

This chapter has the following objectives,

(1) Identify questions to be addressed to test the potential of WSI, through addressing in-depth, the implementation of the WASP associative memory.

(2) Identify experiments that need to be performed to answer the questions identified in (1).

(3) Detail how the data from these experiments will be treated and the conclusions that may be drawn.

5.2 Proposed questions.

The questions to be answered all relate to the engineering of an associative memory for WASP that fulfills the following criteria, namely

(1) What are the design characteristics of associative memories and which are the cost-effective designs?

(2) How well do ASPs with differing associative memory designs meet the technological constraints of WSI?

(3) What implications does the use of the different associative memory designs have on the PE packing density of the WASP device?

5.3 Experimentation

The first question in 5.2 has been addressed by the set of experiments 1-3 as outlined below. The second and third questions have broken down into 2 separate investigations into WSI implementation issues (viz. experiment 4) and PE·packing density (viz. experiment 5).

5.3.1 Experiment 1 - associative memory characteristics

This experiment involved the investigation of the electrical, physical and control characteristics of a range of associative memory designs that meet the WASP functional requirements.

To this end, the experiment identified a range of circuit designs that met the WASP functional requirements. These circuits were laid-out and their physical characteristics (viz. pitch and layout area) determined.

Using these layouts, circuit simulations were performed to identify the electrical characteristics of the designs for each of their functions. Measurements of operation speed and power consumption have been noted and this information was fed back to modify the circuit layout so as to meet the performance requirements (eg. increase transistor geometries to improve speed).

From these physical and electrical characteristics it was possible to identify the control requirements of each design (eg. control signals required, timing constraints, control logic complexity and drive capability). This data was used to expose the control overhead of the bit and word-control logic for each design.

The data from this experiment provided a comprehensive understanding of the design characteristics and tradeoffs for a range of associative memory designs.

5.3.2 Experiment 2 - evaluation in ASP environment

Having obtained details of associative memory characteristics 'in isolation', this experiment aimed to evaluate the performance of the designs within the ASP environment.

This was achieved by calculating the performance of the associative memory designs based on the 'typical' ASP operational cycle. This provided detailed information on the associative memory performance in a realistic operational environment and hence helped better identify promising associative memory designs for WASP.

5.3.3 Experiment 3 - associative memory failure modes

In an investigation into associative memory characteristics for VLSI implementation, the data from Experiments 1 and 2 would be sufficient for a thorough evaluation of designs. However, in WSI another factor, namely the yield (viz. the number of circuit elements expected to be functional) and harvest (viz. the proportion of functional circuit elements that need to be accessible to meet the system requirements).

For example, some designs which have promising electrical characteristics may in reality pose such yield and harvest overheads as to make it infeasible to achieve a sufficient number of wafers with the requisite number of useable elements. Consequently, it is the objective of this experiment to evaluate the APE yield and harvest that would be achieved by the use of the 5 different

associative memory designs in the image processing WASP device.

5.3.4 Experiment 4 - WSI implementation issues

The objective of this experiment is to evaluate the suitability of the previously proposed associative memory designs for WSI implementation.

The major unaddressed WSI implementation issues relevant to associative memories are electrical and physical design issues. Since the previous experiments will have identified designs that meet the speed requirements. The issues that concern WSI electrical and physical design are power dissipation and power supply.

This is investigated by determining the constraints that WSI imposes on power dissipation and power supply and evaluating the performance of a WASP device comprising these designs with respect to these constraints.

5.3.5 Experiment 5 - PE packing density

The objective of this experiment is to determine the influence the different designs has on WASP PE packing density.

5.4 Evaluation of results

The results can be used to evaluate the following,

(1) To identify design characteristics and tradeoffs of a range of associative memories (Experiments 1-3).

(2) Evaluate the suitability of different associative designs for

WSI implementation (Experiment 4).

(3) Determine the influence these designs have on WASP PE packing density (Experiment 5).

From these results it is then possible to draw wider-ranging conclusions relating to

(1) Design constraints of WSI and the implications these constraints have on the engineering of fine-grain parallel processing systems in WSI.

(2) Suitable design techniques for implementing fine-grain WSI parallel processing computer systems.

5.5 Conclusions

This chapter has identified questions that need to be addressed to further investigate the potential of WSI for implementing fine-grain parallel processing computer systems, through a specific investigation into the implementation of the ASP architecture in WSI. To this end, a set of 5 experiments have been proposed to address these questions. These experimental results will be used to draw wider-ranging conclusions as to the potential of WSI for implementing fine-grain parallel processing architectures.

CHAPTER 6. EVALUATION OF ASSOCIATIVE MEMORY DESIGNS

6.1 Objectives

The objectives of this chapter are concerned with Experiments 1 and 2 discussed in the previous chapter, namely

(1) Investigate, identify and evaluate the electrical, physical and control characteristics of a range of associative memory designs that meet the WASP design constraints.

(2) Evaluate the performance of suitable associative designs within the WASP operational environment.

6.2 Associative memory operation.

In concept, an associative memory that operates as part of an associative parallel processor, comprises a 2-dimensional array of content-addressable memory (CAM) cells (Figure 6-1.). The CAM cells share common data lines in the bit-column direction and common match lines in the word-row direction, controlled with supporting bit (BCL) and word (WCL) control logic. In operation, there are 3 types of primitive parallel processing operations that a CAM cell can perform

(1) Searching - where a search pattern is presented by the CAM array BCL to all CAM cells simultaneously. The search pattern presented is compared against the datum content of the CAM cell and a match or mismatch is signalled on the match lines associated with each word. In practice, for each word row, there are usually fewer match lines than bits of CAM. Consequently, match lines are shared between many bits in the

same word-row. This creates a requirement for maskable search arguments to be supported, such that a particular bit-column guarantees a match regardless of the data contents of cells in masked bit-positions.

(2) Writing - unlike most other memory structures, it is possible to have all CAM words active for writing simultaneously (eg. when initialising the data registers in each APE). Indeed, this is a desirable occurrence since it suggests that efficient use is being made of the parallelism available. It is also possible to specify which bit-columns are to be masked out of the write operation, in a manner analagous to masking out bit-columns during search operations.

(3) Reading - this operation involves the transmission of the memory data from the CAM cells to the CAM array BCL. Due to the associative access to the memory (viz. absence of address decoding), it is possible to specify a read from many CAM words simultaneously (although, this usually occurs as due to an algorithmic error, rather than deliberate choice).

6.3 WASP design constraints

The WASP IP device, like its VLSI equivalent, SCAPE [8,9] is targetted towards the support of image processing applications. Consequently, many of the WASP design constraints are based on those of the SCAPE chip.

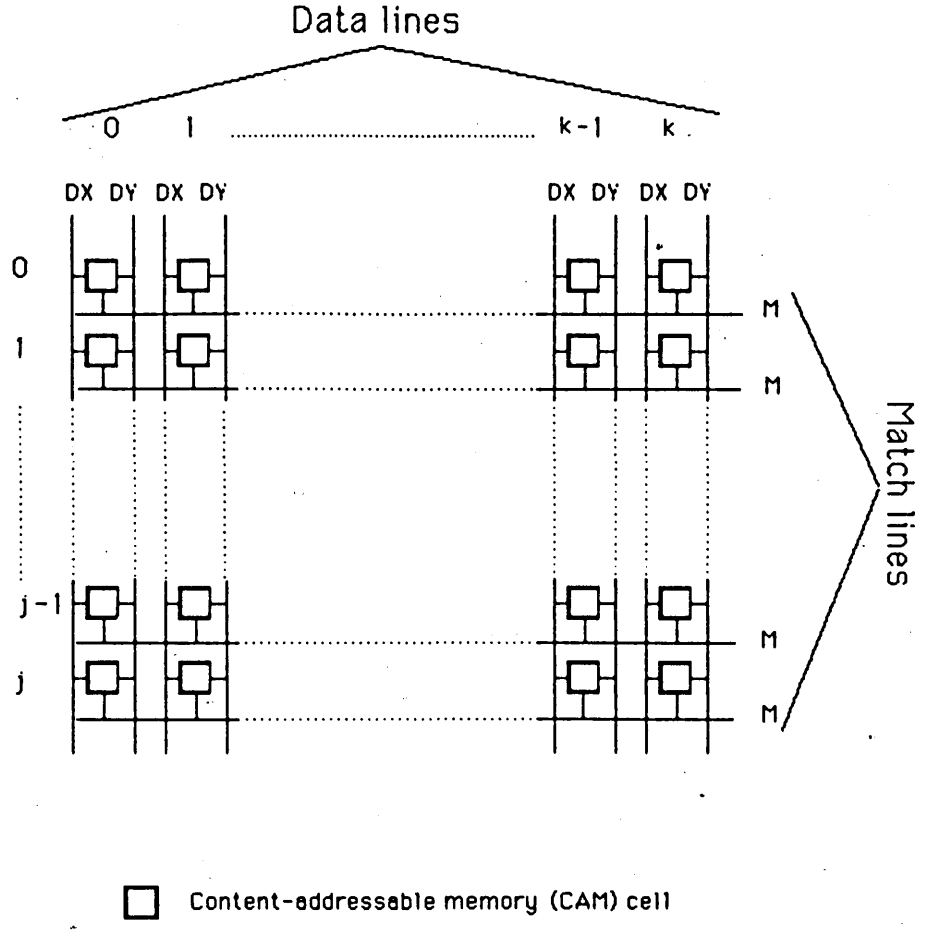


Figure 6-1. Associative memory structure.

6.3.1 Functional design constraints

The following functional requirements have been identified

(1) Bit-column interface - this provides the search and write arguments to the CAM cells. Two data lines are needed to communicate the search and write arguments (namely '0', '1', and 'X' for the masking operation). These lines may also be used to transmit the results of read operations back to the BCL. Alternatively, a dedicated read line may be used for this purpose.

(2) Word-row interface - this receives the result of a search argument from each word-row of CAM. There are 2 match lines per word-row (namely M0 and M1) and all CAM cells per word-row connect to both lines. The M0 line signals if a '0' was matched or mismatched and the M1 line signals if a '1' was matched or mismatched. Each CAM cell in a word row is connected to both the M0 and M1 lines. The use of two match lines per word-row allows two bits of data to be transmitted to the word-row interface simultaneously. This assists the efficient processing of the bit-serial operations that predominate in image and signal processing algorithms.

6.3.2 Operational design constraints

Experience in the design of the SCAPE chip CAM [8,77] suggested that in order to achieve the 25ns processing rate imposed by the 40 MHz clock cycle of the SCAPE chip, then the CAM cell must meet or surpass the following (worst case) performance targets.

Search : < 5ns Write : < 5ns Read : < 5ns

6.3.3 Technological design constraints

The same Plessey 2-micron, 2-layer metal, p-well, bulk-CMOS processing used in the design of the VLSI ASPs SCAPE and SCRIPT and projected for use in the design of the WASP device is used for CAM cell layout. Furthermore, a minimum 'safe' transistor geometry of 2-micron (gate) and 5-micron (channel) is used.

6.4 Investigation strategy - experiment 1

A representative selection of CAM cells have been chosen as candidate cells for investigation and the following actions undertaken

(1) Layouts were generated for each of these cells and the size and layout constraints determined.

(2) Circuit simulations of each of these cells were undertaken (using SPICE with 'typical' transistor models operating at 27°C), aimed at identifying the speed, current and power characteristics for the read write and search operations.

(3) An analysis of the function table and operational characteristics of the CAM designs was performed to expose control considerations.

6.5 Candidate CAM designs

Five different designs are evaluated, ranging from a simple pseudo-static design to a sophisticated, potentially low-power and easily controllable CAM. However, all five designs have common features.

6.5.1 Match logic

The requirements dictate two match lines shared between all CAM cells per word row (namely the M0 and M1 lines). There appear to be 3 main design techniques for achieving this,

(1) Individual match lines - here each CAM cell has its own match lines and the combination of these lines to form composite M0 and M1 signals is performed by the word-row control logic. This scheme offers maximum flexibility, but requires many match lines to travel across the CAM word. For all but the shortest word width, this may make the CAM cell dimensions too large.

(2) Recursive match logic - here each CAM cell computes the match function (viz. match or mismatch) as a function of the match data from the previous CAM cell and its own data and match arguments. The result of this function is then sourced to the next CAM cell in line. This approach reduces the number of lines, but effectively requires a signal to ripple from one end of the word to the word-row control logic. As each stage will involve at least one transistor in series, for all but the shortest word widths, the resulting delay may prove unacceptable.

(3) Wire AND match lines - here the M0 and M1 outputs from each CAM cell are wire ANDed together. Initially, each match line is precharged high and is conditionally discharged should the data stored in any one CAM cell in a word row mismatch the search argument. If a '0' is being searched for (and the CAM cell stores a '1') then M0 is discharged. If a '1' is being searched

for (and the CAM cell stores a '0') then M1 is discharged.

This approach requires only two lines to be passed across the CAM cells. In addition, the precharge/discharge mechanism is fast and relatively simple to implement.

Consequently, the wire AND approach is used to realise the match logic in all the following designs.

6.5.2 Data storage logic

There are effectively 3 circuit design options available, namely

(1) Static - data is stored through 2 two cross-coupled inverters. Once data is stored, it will remain there as long as power is supplied.

(2) Self-refreshing or Pseudo-static - here data storage occurs through making use of the capacitance on the transistor gates. This charge will decay in time but by asserting a control signal, the cell can be configured to behave like an static storage device and restore the data signals at the expense of increased power consumption.

(3) Dynamic - akin to the 3 and 1 transistor RAM cells, this design is compact but relies on external logic to sense and refresh the data signals.

The static data store is the most 'rugged' circuit option. The pseudo-static cell is simpler, but requires the data to be regularly refreshed. However, its self-refreshing capability means that all cells can be refreshed simultaneously. The dynamic cell structure offers the potential for the highest packing density. However, the

need to regularly refresh each cell individually, does involve complex support logic and a significant time penalty (cf. the relative access times of static and dynamic RAMs), which could seriously slow down the operation of a 40 MHz parallel processor. More importantly, since both data (D) and the inverse data ($\overline{D}$) needs to be held in each cell (to support the match logic) two dynamic storage circuits per CAM would be needed.

Consequently, only static and pseudo-static circuit designs have been selected for investigation.

6.6 Candidate CAM circuits

This section details the structure and operation of the CAM designs

6.6.1 CAM A - pseudo-static nMOS CAM

This cell (Figure 6-2. and Table 6-1) is the simplest design considered (in terms of number of transistors). It comprises only n-channel devices and the data storage part of the CAM cell is pseudo-static, in that when the RW line is driven high, the cell acts as a static nMOS flip-flop. However, driving the RW line to ground, conserves power by relying on dynamic charge for data storage. Consequently, this cell needs to be refreshed frequently if CAM data is not to be lost.

Writing a '0' or '1' to the cell is accomplished by driving the Dx and Dy lines to complementary data values and strobing the RW line. A masked write (and refresh) operation is achieved by driving both data lines high and strobing the RW line. This causes the cell to behave as a static nMOS flip-flop.

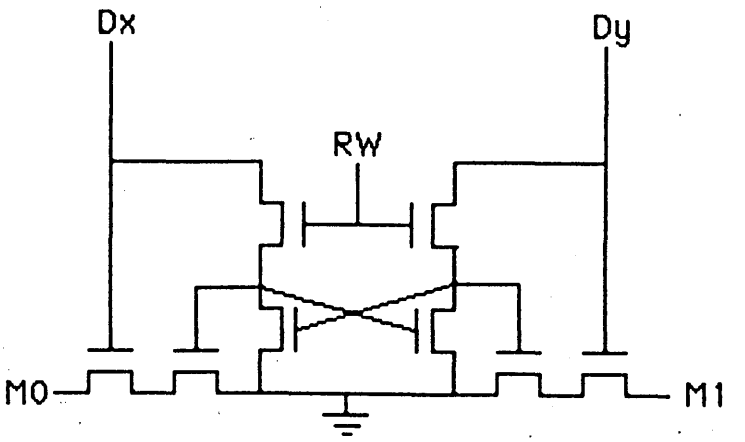
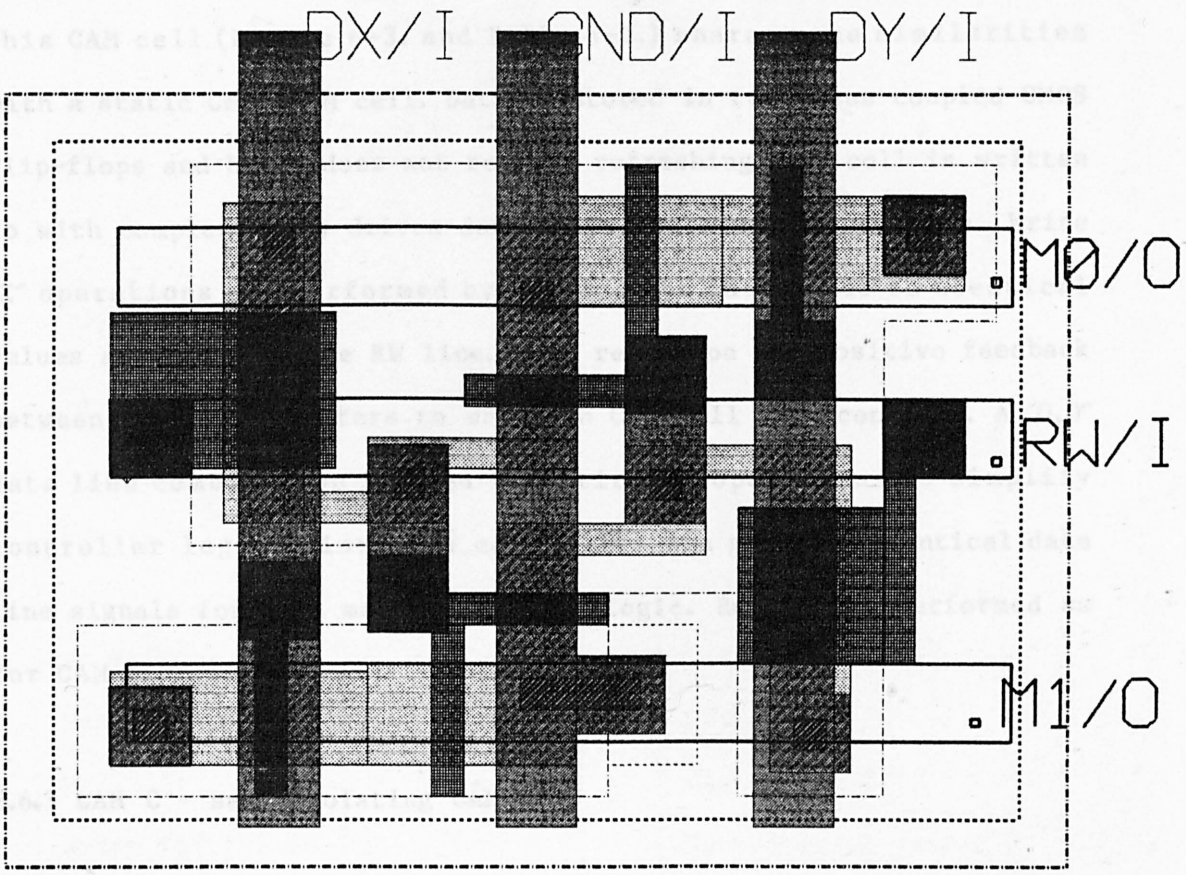


Figure 6-2. CAM A - pseudo-static nMOS CAM.

Function	Dx	Dy	RW	M0	M1	Comment
No Write	X	X	0	X	X	CAM data unchanged
Write "0"	0	1	1	X	X	CAM stores "0"
Write "1"	1	0	1	X	X	CAM stores "1"
Write "X"/	1	1	1	X	X	Masked write on CAM
Refresh						CAM self-refreshes
Match "0"	1	0	0	P	P	M0 -> 0 if CAM = "1"
Match "1"	0	1	0	P	P	M1 -> 0 if CAM = "0"
Match "X"	0	0	0	P	P	M0/1 unchanged
No Read	P	P	0	X	X	Dx/y unchanged
Read	P	P	1	X	X	Dx/y = CAM data

Table 6-1. Function table for CAM A.



CAM A - layout.

Reading is performed by precharging Dx and Dy high. Driving RW high will then cause one of the data lines to preferentially discharge, reflecting the CAM cell data contents. This voltage differential can then be sensed and amplified in the BCL.

6.6.2 CAM B - static CMOS CAM

This CAM cell (Figure 6-3. and Table 6-2.) shares some similarities with a static CMOS RAM cell. Data is stored in two cross coupled CMOS flip-flops and hence does not require refreshing. The cell is written to with complementary driven data lines and a strobed RW line. Write 'X' operations are performed by driving the data lines to identical values and strobing the RW line. This relies on the positive feedback between the two inverters to maintain CAM cell data contents. A '0,0' data line combination is used for write 'X' operations to simplify controller logic, since the controller can generate identical data line signals for both match and write logic. Reading is performed as for CAM A.

6.6.3 CAM C - self-isolating CAM

This cell (Figure 6-4. and Table 6-3.) is a development of CAM B and has the same function table. Where this CAM cell differs from CAM B is in the isolation of the data storage part of the CAM cell from the Dx and Dy lines during write 'X' operations. This avoids the need to rely on the positive feedback between the cross-coupled inverters to maintain the data contents of the CAM cell and hence reduces write 'X' power consumption. Reading is performed as for CAMs A and B.

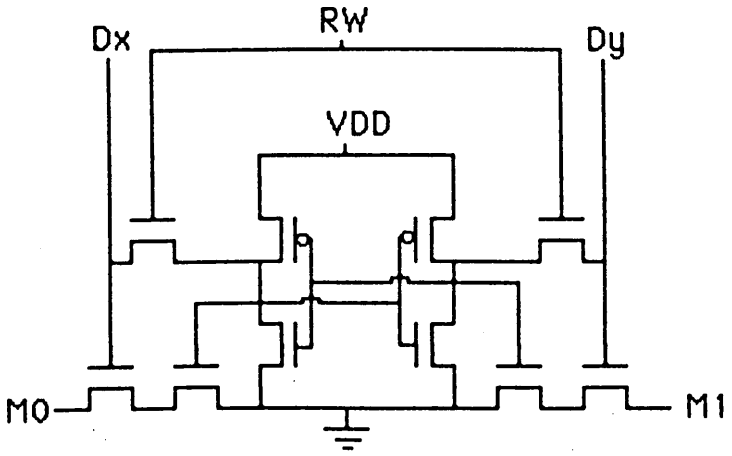
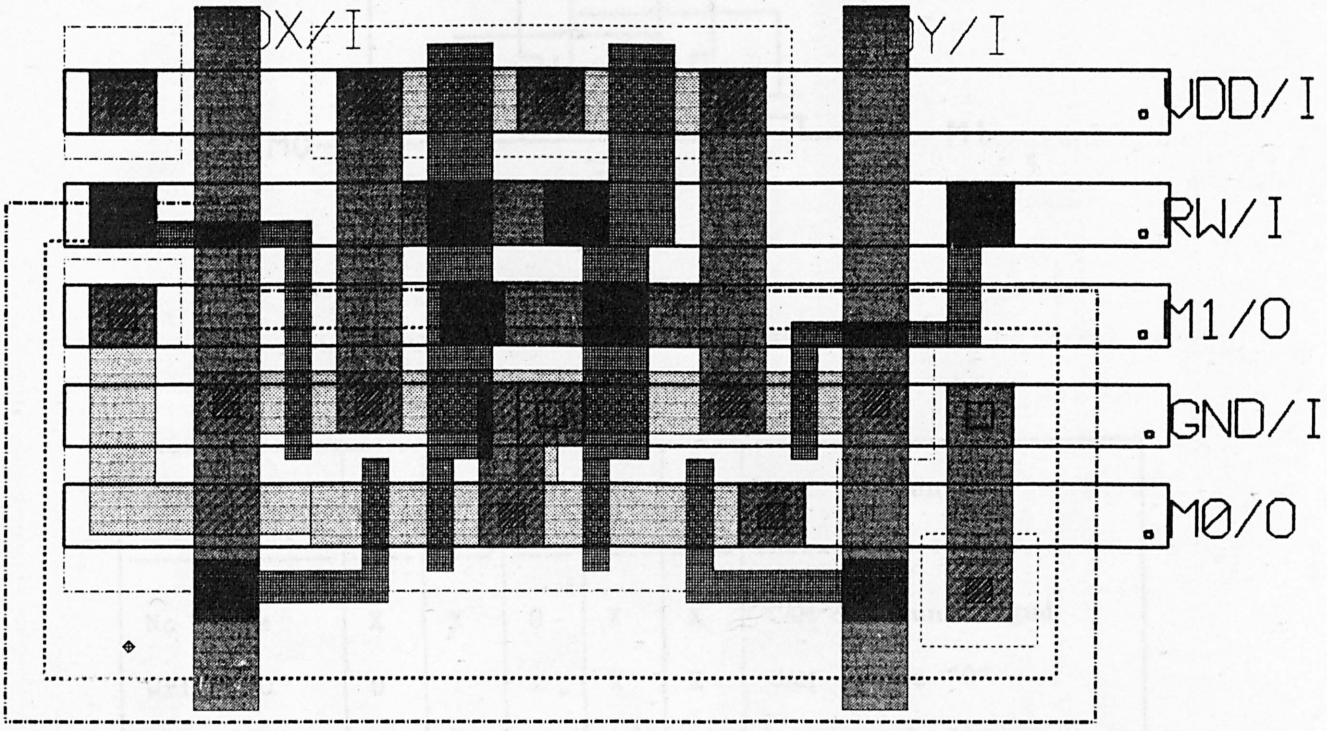


Figure 6-3. CAM B - Static CMOS CAM.

Function	Dx	Dy	RW	M0	M1	Comment
No Write	X	X	0	X	X	CAM data unchanged
Write "0"	0	1	1	X	X	CAM stores "0"
Write "1"	1	0	1	X	X	CAM stores "1"
Write "X"	0	0	1	X	X	Masked write on CAM
Match "0"	0	1	0	P	P	M0 -> 0 if CAM = "1"
Match "1"	1	0	0	P	P	M1 -> 0 if CAM = "0"
Match "X"	0	0	0	P	P	M0/1 unchanged
No Read	P	P	0	X	X	Dx/y unchanged
Read	P	P	1	X	X	Dx/y = CAM data

Table 6-2. Function table for CAM B.



CAM B - layout.

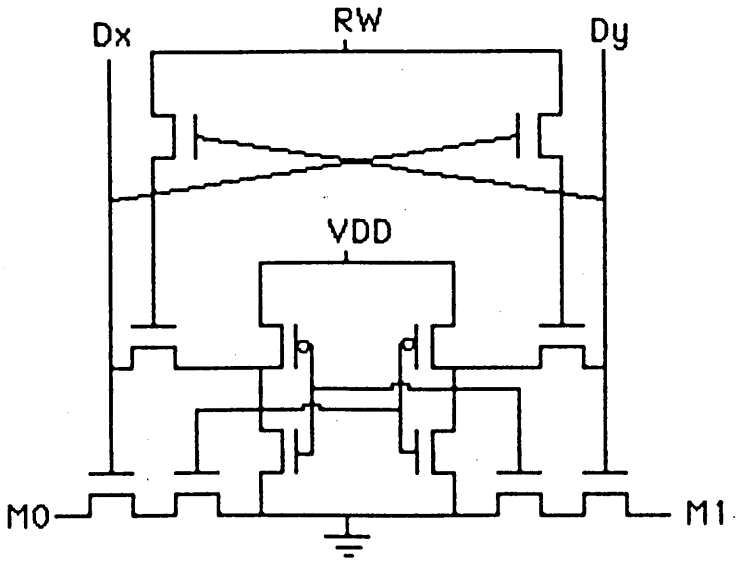
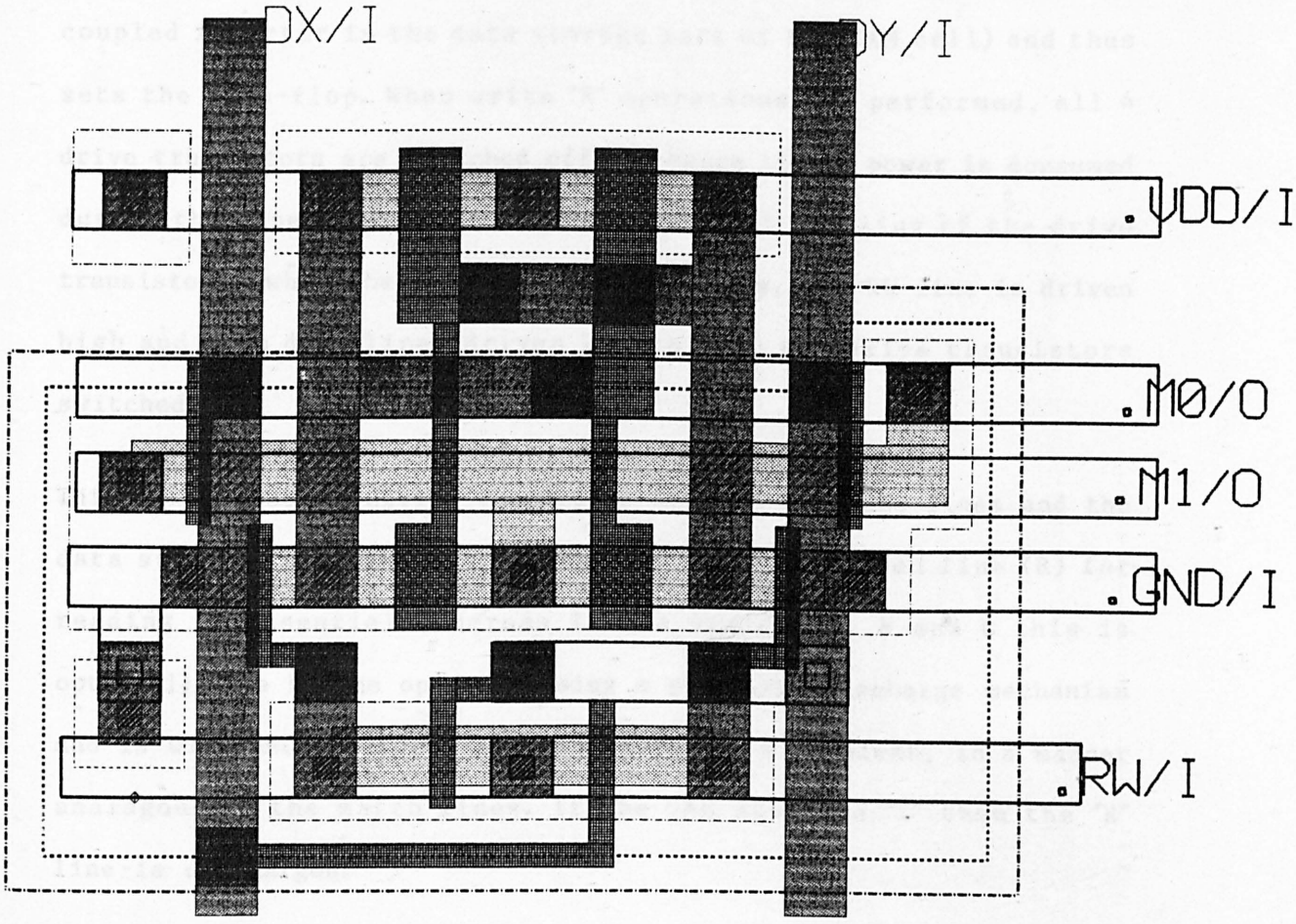


Figure 6-4. CAM C - Self-isolating CAM.

Function	Dx	Dy	RW	M0	M1	Comment
No Write	X	X	0	X	X	CAM data unchanged
Write '0'	0	1	1	X	X	CAM stores '0'
Write '1'	1	0	1	X	X	CAM stores '1'
Write 'X'	0	0	1	X	X	Masked write on CAM
Match '0'	0	1	0	P	P	M0 -> 0 if CAM = '1'
Match '1'	1	0	0	P	P	M1 -> 0 if CAM = '0'
Match 'X'	0	0	0	P	P	M0/1 unchanged
No Read	P	P	0	X	X	Dx/y unchanged
Read	P	P	1	X	X	Dx/y = CAM data

Table 6-3. Function table for CAM C.



CAM C - layout.

6.6.4 CAM D - capacitive load CAM

The other static CAM designs (B, C) have used a conventional resistive load to write data. This design (Figure 6-5. and Table 6-4) presents a capacitive loading to the data lines. Toggling of the CAM D design is accomplished by setting the appropriate pattern on the data lines and strobing the RW line. This charges up 2 of the 4 'drive' transistors (viz. the transistors in series with each cross-coupled inverter in the data storage part of the CAM cell) and thus sets the flip-flop. When write 'X' operations are performed, all 4 drive transistors are switched off and hence little power is consumed during this operation. To prevent accidental charging of the drive transistors, when the CAM array is on standby, the RW line is driven high and both data lines driven low to keep the drive transistors switched off.

This design has no direct connection between the data lines and the data store in the CAM cell. Consequently, a dedicated line (R) for reading is essential (whereas in the designs A, B and C this is optional). The R line operates using a precharge/discharge mechanism and is wire ANDed between all CAM cells per bit-column, in a manner analagous to the match lines. If the CAM stores a '1' then the 'R' line is discharged.

6.6.5 CAM E - 'simple' capacitive load CAM

CAM E is a simplified version of CAM D (Figure 6-6. and Table 6-5). CAM E has only two drive transistors and relies on the positive feedback between the two cross-coupled inverters in the data part of the CAM cell to complete the write operation. The read operation is as for CAM D

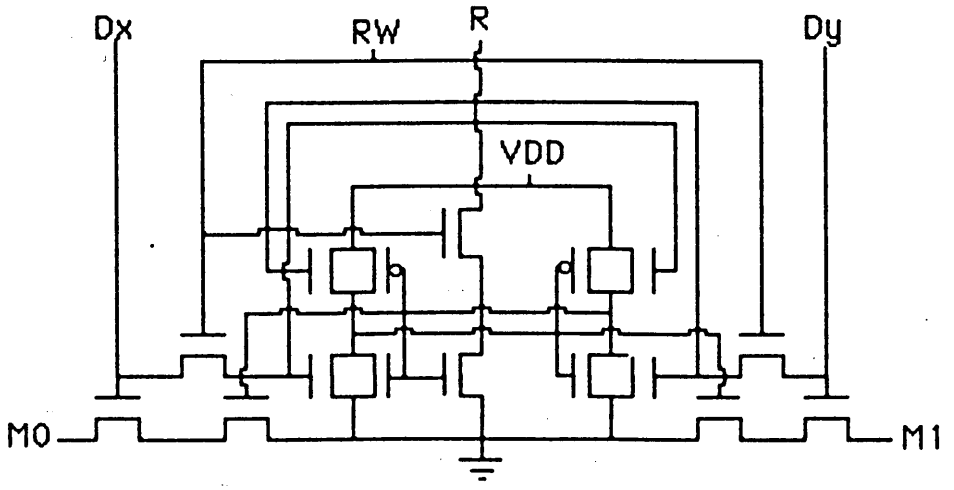
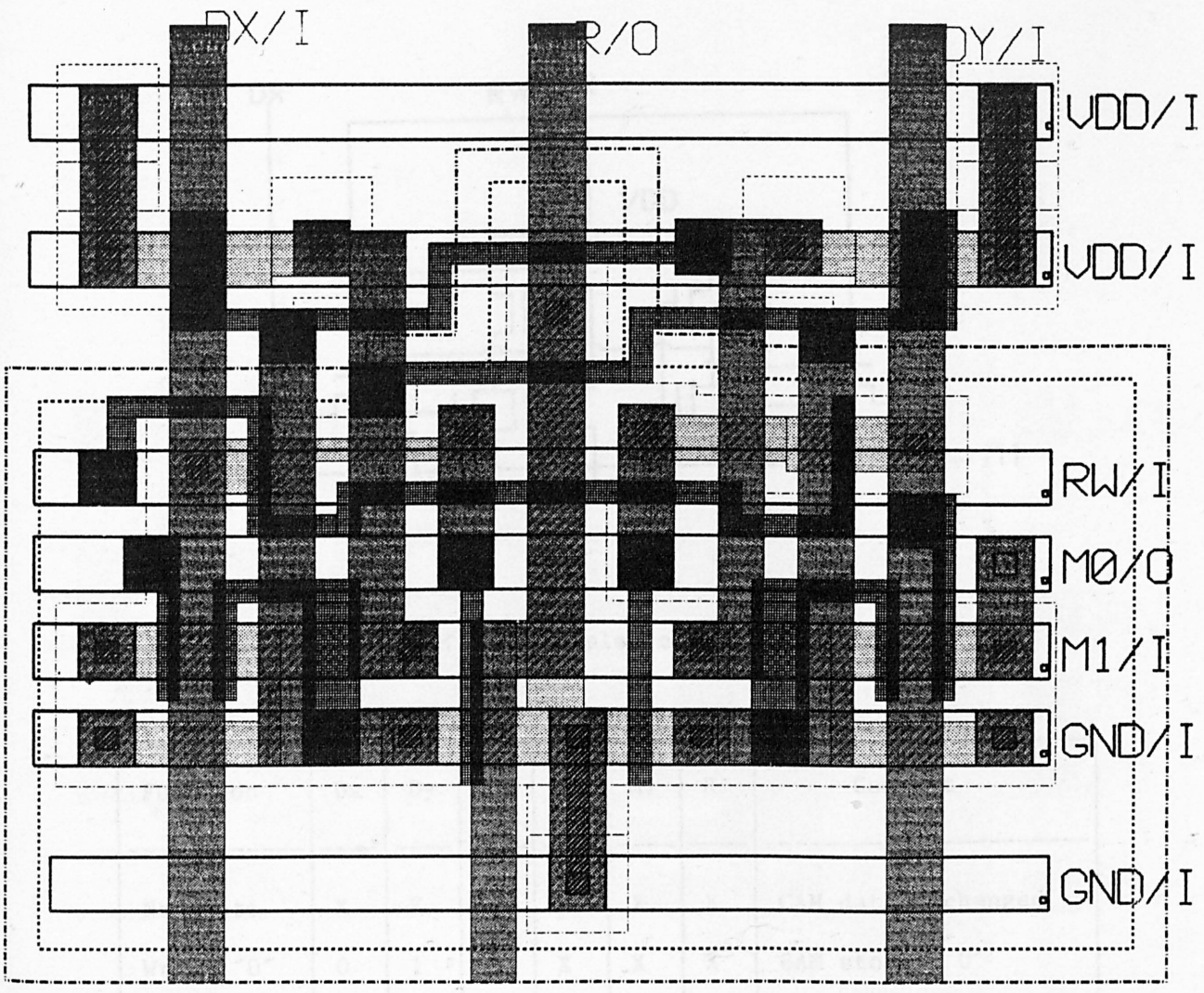


Figure 6-5. CAM D - Capacitive load CAM.

Function	Dx	Dy	RW	M0	M1	R	Comment
No Write	X	X	0	X	X	X	CAM data unchanged
Write "0"	0	1	1	X	X	X	CAM stores "0"
Write "1"	1	0	1	X	X	X	CAM stores "1"
Write "X"/	0	0	1	X	X	X	Masked write on CAM
Standby							CAM standby mode
Match "0"	0	1	0	P	P	X	M0 -> 0 if CAM = "1"
Match "1"	1	0	0	P	P	X	M1 -> 0 if CAM = "0"
Match "X"	0	0	0	P	P	X	M0/1 unchanged
No Read	0	0	0	X	X	P	R unchanged
Read	0	0	1	X	X	P	R = CAM data

Table 6-5. Function table for CAM D.



CAM D - layout.

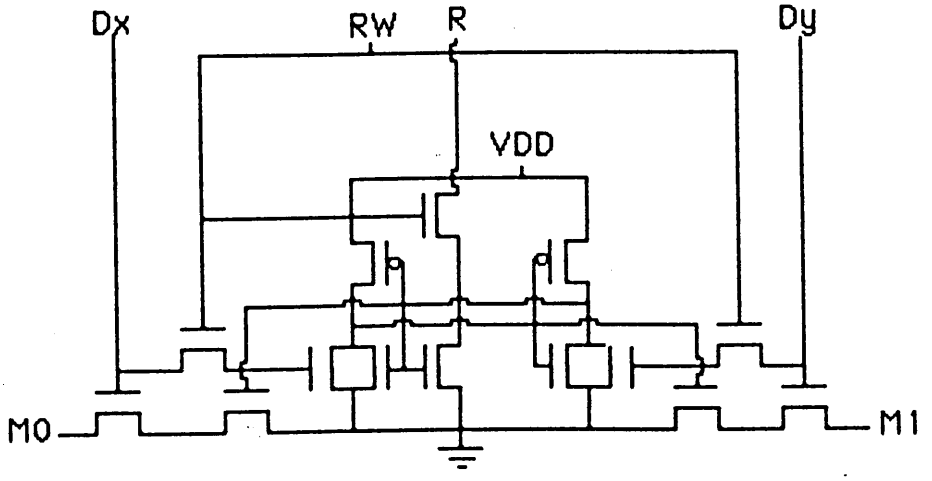
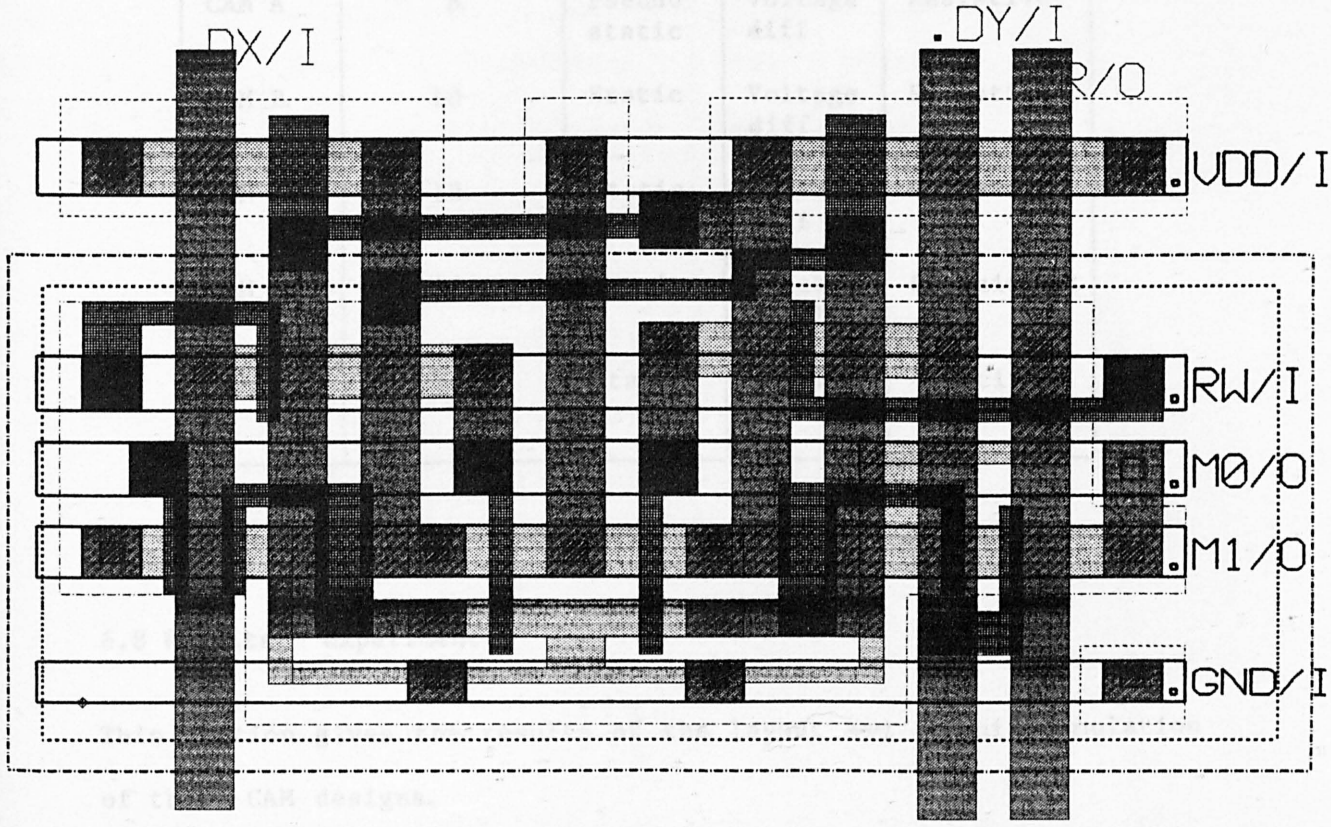


Figure 6-6. CAM E - 'Simple' capacitive load CAM.

Function	Dx	Dy	RW	MO	M1	R	Comment
No Write	X	X	0	X	X	X	CAM data unchanged
Write '0'	0	1	1	X	X	X	CAM stores '0'
Write '1'	1	0	1	X	X	X	CAM stores '1'
Write 'X'/	0	0	1	X	X	X	Masked write on CAM
Standby							CAM standby mode
Match '0'	0	1	0	P	P	X	MO -> 0 if CAM = '1'
Match '1'	1	0	0	P	P	X	M1 -> 0 if CAM = '0'
Match 'X'	0	0	0	P	P	X	MO/1 unchanged
No Read	0	0	0	X	X	P	R unchanged
Read	0	0	1	X	X	P	R = CAM data

Table 6-6. Function table for CAM E.



CAM E - layout.

6.7 Summary

A range of CAM designs have been proposed with different read and write mechanisms. Table 6-6. summarises these designs.

Name	Transistors	Data storage	Read	Write
CAM A	8	Pseudo-static	Voltage diff	Resistive
CAM B	10	Static	Voltage diff	Resistive
CAM C	12	Static	Voltage diff	Resistive
CAM D	16	Static	Prechg/ dischg	Capacitive
CAM E	14	Static	Prechg/ dischg	Capacitive

Table 6-6. Comparison of CAM designs.

6.8 Results - experiment 1

This section gives the results of the layout and circuit simulation of the 5 CAM designs.

6.8.1. Layout area

The different designs have widely differing layout areas. In part, this is due to the number of transistors in each design. However, the area occupied by the p-well proved to be a major layout constraint. Indeed, even if only 1 or 2 p-channel transistors are used, the area occupied by the p-well is close to half the total layout area. This p-well overhead is caused by the need to maintain a relatively large (> 10 um) separation between wells.

Table 6-7. summarises the layout characteristics of the 5 CAM designs, where relative area is defined as

$$\frac{\text{Area of CAM cell}}{\text{Area of CAM A}}$$

Name	Transistors			Area	Relative
	P	N	Total	(uM^2)	area
CAM A	0	8	8	1944	1.00
CAM B	2	8	10	2980	1.53
CAM C	2	10	12	4160	2.14
CAM D	2	14	16	5500	2.82
CAM E	2	12	14	5240	2.69

Table 6-7. CAM cell layout experimental results.

6.8.2 Speed of operation

It was found relatively easy to achieve the 5ns match/read/write speed targets with all designs. Furthermore, there was relatively little to choose between the designs in terms of speed. Consequently, the analysis concentrates on CAM power and current.

6.8.3 Current consumption

The search and read operations being based on a precharge/discharge mechanism consume relatively little current. However, it is during the write operations that the designs exhibit widely different current requirements.

Table 6-8., summarises the maximum current drawn by the CAM cells during write operations (where '0/1' and 'X' refers to the particular type of write operation).

Name	Current (uA)	
	(0/1)	(x)
CAM A	71.0	104.0
CAM B	175.1	117.2
CAM C	122.3	<1
CAM D	168.9	<1
CAM E	134.4	<1

Table 6-8. CAM cell current data.

Due to the lower switching currents in N-channel circuits allied to the fact that data '1' in CAM A is represented by 3.2V and hence the pull-down transistors in the data part of CAM A are not fully on, CAM A has a significantly lower switching current than the CMOS designs (viz. those designs incorporating both 'n' and 'p' transistors). Furthermore, design A is the only cell that has a higher write 'X' current than a write '0/1' current. Interestingly, the high switching currents of CMOS circuits and the fact that the pull-down transistors in the data part of the CAM cell are fully on means that the static CMOS design B has higher write '0/1' and 'X' currents than the nMOS design.

The designs that incorporate special-purpose logic to reduce write 'X' currents (CAMs C, D and E) succeed with this goal, and all have very low write '0/1' current figures.

6.8.4 Power consumption

The power consumption of the designs, track fairly closely with the figures for current drawn. Table 6-9., summarises the power consumption for each of the designs performing write '0/1' and write 'X' operations at 40 MHz.

Name	Power (uW)	
	(0/1)	(x)
CAM A	51.0	99.8
CAM B	149.0	120.6
CAM C	165.4	<1
CAM D	172.8	<1
CAM E	173.8	<1

Table 6-9. CAM cell power data (operating at 40 MHz).

From the above table it can be seen that the CMOS (viz. designs B - E) CAM's have approximately the same power consumption during write '0' and write '1' operations. CAM C has a higher power consumption than CAM B since the write 'X' isolation logic results in the CAM cell being written to from 1 side only, slowing the write operation down slightly and hence prolonging the period during which current is drawn. CAMs D and E which use a capacitive loading technique to toggle the flip-flop, have similar power figures. The slightly higher current peak of CAM D being compensated by its marginally faster write speed.

CAM A has a significantly lower power figure than the CMOS designs,

this due to the combination of lower switching currents in nMOS circuitry and a relatively fast write speed being achieved through the complementary-driven data lines supplying the power to the cell during write operations.

CAMs C, D and E by virtue of their circuit designs, consume very little power during write 'X' operations. CAM A consumes more power during write 'X' than during write '1' or write '0' operations. However, the high dynamic currents inherent to CMOS designs results in CAM B consuming the most power during write 'X' operations.

6.9 Observations - experiment 1

During the design and evaluation of the performance of the CAM designs, 2 observations were made as to the behaviour of associative memories which mark them out from other memory forms.

6.9.1 2-phase write

Conventional RAM memories have timing constraints which require the data to be valid prior to the RW line being driven inactive. However, with associative memories, the existence of a write 'X' operation, complicates the operation.

If when the RW line is driven high, the cell holds a '1' (say) but the data lines are initially registering a '0' (say) pattern, then a '0' is temporarily stored in the cell. With RAMs this would be acceptable since as long as the correct value arrived on the data lines prior to the RW line going inactive, the correct value would eventually be stored in the memory. However, in the same circumstances with CAMs, if the final state of the data lines is 'X',

then the following sequence of actions occurs,

(1) Data lines = '0', RW line inactive, CAM data = '1'.

(2) Data lines = '0', RW line active, CAM data = '0'.

(3) Data lines = 'X', RW line active, CAM data = '0'.

(4) Data lines = 'X', RW line inactive, CAM data ='0'.

Thus, the presence of the 'X' state requires the data lines to be valid, prior to the RW line being active. This requires that CAM writes be 2-phase operations.

6.9.2 Multi-read

A consequence of the associative access to CAMs is that any number of CAM words can be selected for reading. This can cause some problems with those cells that do not have a dedicated read line (eg. CAMs A, B and C). If for example, a large number of words are selected for reading simultaneously, and for 1 particular bit-column all cells but 1 contain a '0' (say) and the other a '1', then the effect of reading from all cells simultaneously is to drive the data lines to the write '0' state. To the one cell that stores a '1', the data lines now flip it over into the '0' state, corrupting cell contents.

While it is difficult to conceive of a situation occuring where a multi-word read would be deliberately specified, it is possible that this could occur as a program error and thus offers the potential for subtle data corruption to occur.

Those designs (eg. CAMs D - E) that use a dedicated wire AND read line, do not suffer from this problem and thus in this respect, may

be considered to be more reliable.

6.10 Analysis of control requirements

This section uses the information gained in the circuit simulation of the CAM designs and an analysis of the function tables of the designs, to expose their control requirements.

6.10.1 CAM A

When writing data to the memory, CAM A presents a resistive load to the bit-column control logic. Consequently, as the number of cells to be driven increases (and it is possible for all cells in a column to be written to simultaneously), the drivers have to become increasingly more poweful in order to be have the potential to toggle an entire bit column. However, CAM A presents a resistive load only to a high-driven line. A low-driven line need only discharge the capacitance held on the data storage gates of the CAM's. Consequently, while powerful pull-up drivers may be needed, modest pull-down drivers will suffice.

An analysis of the control operations (Table 6-1.) shows that the patterns on the data lines for write and search operations are inverted (eg. to write 'X', requires a '1,1' data pattern, but to search 'X' demands a '0,0' pattern). Consequently, an extra BCL logic stage is needed to invert the data line patterns depending on whether a match or search operation is performed.

6.10.2 CAMs B and C

CAM B presents a resistive load to the data lines during writing, for

both high-driven and low-driven lines. Consequently, powerful pull-up and pull-down (cf. CAM A) drivers may be required.

CAM C presents a resistive load only to a low-driven line and thus requires powerful pull-down loads, but only modest pull-up devices.

CAMs B and C have identical search and write data patterns and no requirement for refreshing, thus simplifying bit-column control logic.

6.10.3 CAMs D and E

CAMs D and E present a capacitive load to the data lines during writing. Consequently, only modest buffer sizes are needed. Moreover, the use of intermediary 'drive' transistors in each CAM, allows a data pattern to be strobed into the cell with a short RW pulse. The pattern being stored on the gates of the drive transistors, will then toggle the flip-flop of its own accord. (cf. the other CAMs where the RW line must be on long enough for the cell to toggle).

CAMs D and E have identical search and write patterns, thus simplifying bit-column control logic.

6.10.4 Summary of CAM control requirements

CAMs D and E with their capacitive drive require the smallest drive buffers. CAM A is unique in having different write and search patterns and hence requires extra logic bit-column control logic to support this. Both pseudo-static CAMs need logic to support the refresh operation. CAMs D and E require only a short RW pulse.

6.11 Investigation strategy - experiment 2

The second experiment builds on the data obtained in Experiment 1, by using it to help forecast the performance of different CAM designs within the expected operational environment.

This section evaluates the performance of the CAM cells within the WASP IP APP environment.

6.11.1 Assumptions

The WASP IP APP, like the SCAPE APP, is designed to support image processing algorithms and hence, many of the assumptions are based on information gained in the design of the SCAPE [8,9] chip.

(1) The APP works on a 100ns (40 MHz) instruction cycle, divided up into 4, 25ns 'beats'. Furthermore, control requirements restrict new data arguments to be presented every other beat.

(2) Typical power is based on an analysis of the bit-serial instructions [8] that predominate in ASP image processing algorithms. This figure is derived from CAM power consumption during a 'representative' 100ns instruction of 4, 25ns beats, namely

search : clear : activate/refresh : write

where

(a) Search is a pattern matching operation.

(b) Clear involves write 'X' to all except one bit column (in which 50% of CAM cells toggle).

(c) Activate/refresh selects a new set of active APEs (and in

the case of pseudo-static CAMs refreshes the memory).

(d) Write, updates 18 of the 37 bit-columns (of which 50% toggle) and writes 'X' to the other rows.

6.11.2 Experimental results - experiment 2

Table 6-10. summarises CAM cell power for the 5 CAM cell designs performing typical image processing algorithms.

It can be seen from Table 6-10, that the different CAM designs exhibit a wide range of power consumption figures when operating within the algorithmic and architectural environment of an image processing WASP system. Furthermore, little correlation exists between the CAM power figures when operating in isolation (Table 6-9.) and when acting as part of the WASP device (Table 6-10.).

Name	CAM power (uW)
CAM A	64.7
CAM B	53.6
CAM C	10.6
CAM D	11.1
CAM E	11.2

Table 6-10. CAM power within WASP operational environment.

CAM A which has the lowest write '0' and write '1' power consumption in isolation, has the highest power consumption of all 5 CAMs when

operating in the WASP environment. This is caused by the predominance of write 'X' operations and the need to refresh the memory at regular intervals. The static CMOS CAM B which intuitively one would expect to have a low power consumption, consumes almost as much power as CAM A. This is caused by the combination of the parallel processing operation of CAMs (a high proportion of circuits are active at any one time), the fast clock speed of WASP (40 MHz) and the high switching currents in CMOS circuits.

The more 'sophisticated' CAMs C, D and E overall, have exceptionally low power consumption, since their minimal write 'X' power consumption means that they effectively only consume power when the relatively infrequent (in ASP image processing) write '0' and write '1' operations are performed.

There is of course, a penalty to pay for the reduced power consumption, in this case it is a larger CAM size due to the increased complexity of the CAM cell. This will have an effect on APP size and hence the number of APEs that can be integrated on a wafer. It is the purpose of Experiments 4 and 5 to investigate these effects.

6.12 Use of designs in a VLSI chip

The design of the 256-APE VLSI chip SCAPE, progressed in tandem with the evaluation of the CAM designs. Consequently, it was possible to test out the results of Experiments 1 and 2 by incorporating one of the designs in the SCAPE chip.

Each of the 256 APEs in the SCAPE chip comprises a 37-bit CAM word, resulting in 9472 CAM cells in total. Choice of the cells was

relatively simple, despite all designs (Table 6-11.) meeting the speed limits and being below the pack dissipation limit of 1W. It was found early on, that the major constraint on the design was die size as preliminary estimations suggested that the SCAPE chip size was very close to the well cavity of the 68-pin chip carrier. Increasing well size would have meant using a much more expensive 84 or 100 pin pack. Consequently, design A was selected for SCAPE implementation. Even so, the pressure on chip size was such as to require modifications to be made to the design. In order to reduce the cell size, the minimum drawn transistor channel size was reduced to 3um. While this was still within the design rules, it did mean that processing spreads could vary the actual channel length significantly. Figure 6-7. gives a plot of the final CAM cell layout.

To speed up system operation, the 2-phase write constraint previously discussed was compressed into a 1-phase operation by delaying the RW pulse until the end of the clock phase, thus giving the data lines time to settle. Insufficient area was available to incorporate a seperate R line to avoid multi-read problems. Consequently, users are required to ensure only 1 APE is active during a read operation. Table 6-12. gives a summary of the CAM cell performance. A detailed analysis of the CAM cell in the SCAPE chip by Jones, Jalowiecki, Hedge and Lea can be found in [77].

Figures 6-8 and 6-9 give the circuit diagram and layout of the bit-column control logic for the SCAPE chip that controls a 128-CAM cell column. The dimensions of the cell are

width : 36 um length : 710um.

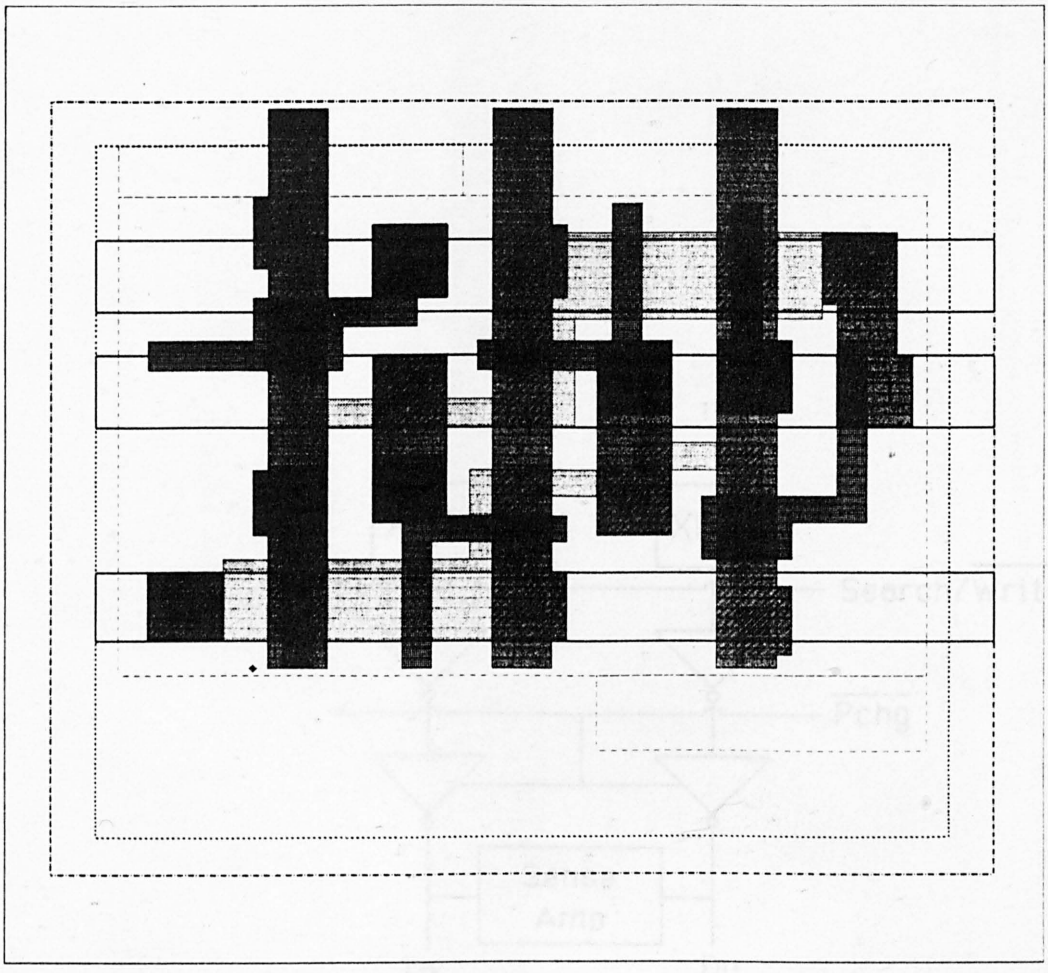


Figure 6-7. Plot of CAM cell selected for SCAPE chip.

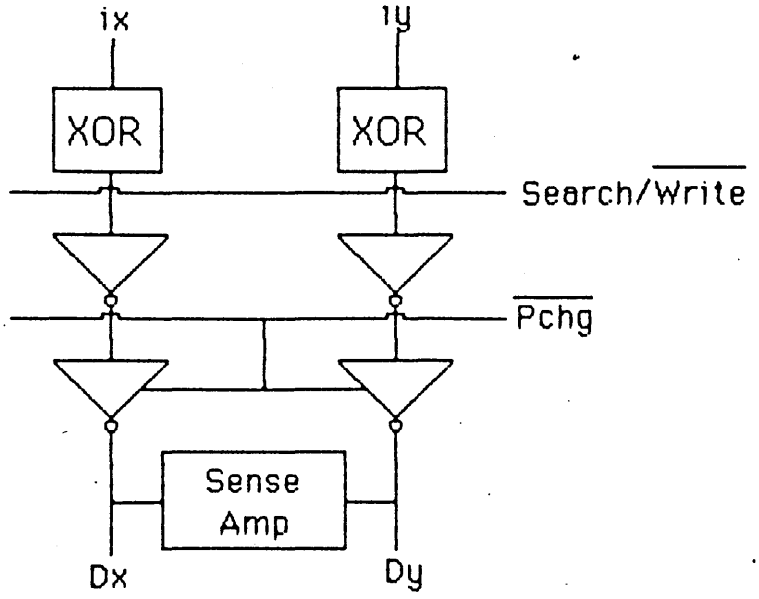


Figure 6-8. Circuit diagram of SCAPE chip BCL.

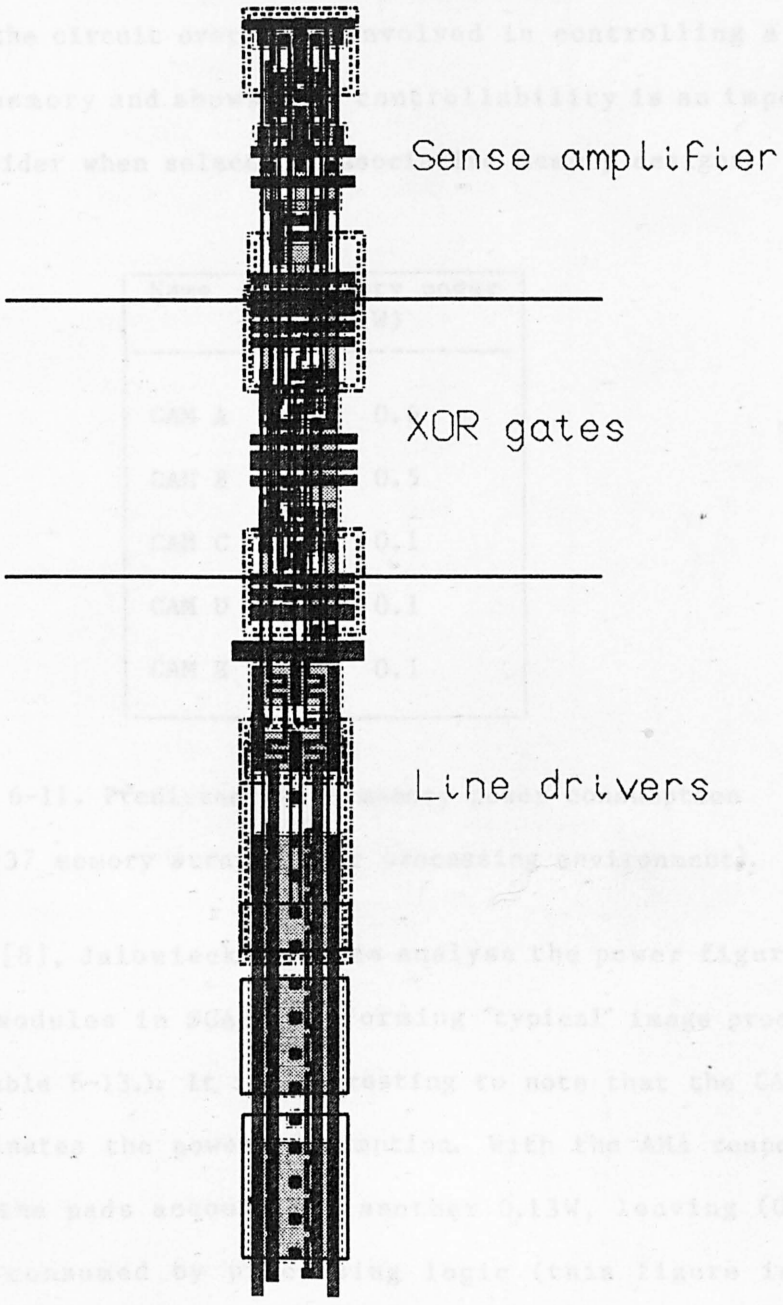


Figure 6-9. Layout of SCAPE chip BCL.

As can be seen, the drivers required to supply current to the resistive load CAM A can be very large. Indeed, over 50% of the area of this logic is occupied by the drive buffers. This clearly demonstrates the circuit overheads involved in controlling a large associative memory and shows that controllability is an important aspect to consider when selecting associative memory designs.

Name	Memory power (W)
CAM A	0.6
CAM B	0.5
CAM C	0.1
CAM D	0.1
CAM E	0.1

Table 6-11. Predicted SCAPE memory power consumption (256 x 37 memory array, image processing environment).

In reference [8], Jalowiecki and Lea analyse the power figures for the various modules in SCAPE performing 'typical' image processing operations (Table 6-13.). It is interesting to note that the CAM cell in SCAPE dominates the power consumption. With the AMA responsible for 370 mW, the pads account for another 0.13W, leaving (0.865 - 0.13) 0.735W consumed by processing logic (this figure is more relevant to WSI as the WASP APP modules do not have to drive off-chip capacitive loads). Therefore the AMA is responsible for (0.37/0.735 x 100) = 51% of the total chip processor power consumption. However, the AMA occupies (Table 6-14.) only (17.5/75 x 100) = 22% of the total chip area, making it by far, the most power dense block in the APP.

SCAPE CAM performance	
Transistors	8
Height	36um
Width	46um
Area	$1656um^2$
Match	5ns
Write	5ns
Read	5ns
Refresh	5ns
Power	44uW
Data '1'	3.2V
Data '0'	0.0V

Table 6-12. SCAPE CAM cell performance.

Module	Power (mW)
BCL	60
WCL	125
AMA	370
MOGL	180
Pads	130
Total	865

Table 6-13. SCAPE chip power by module (from [8])
(image processing operations).

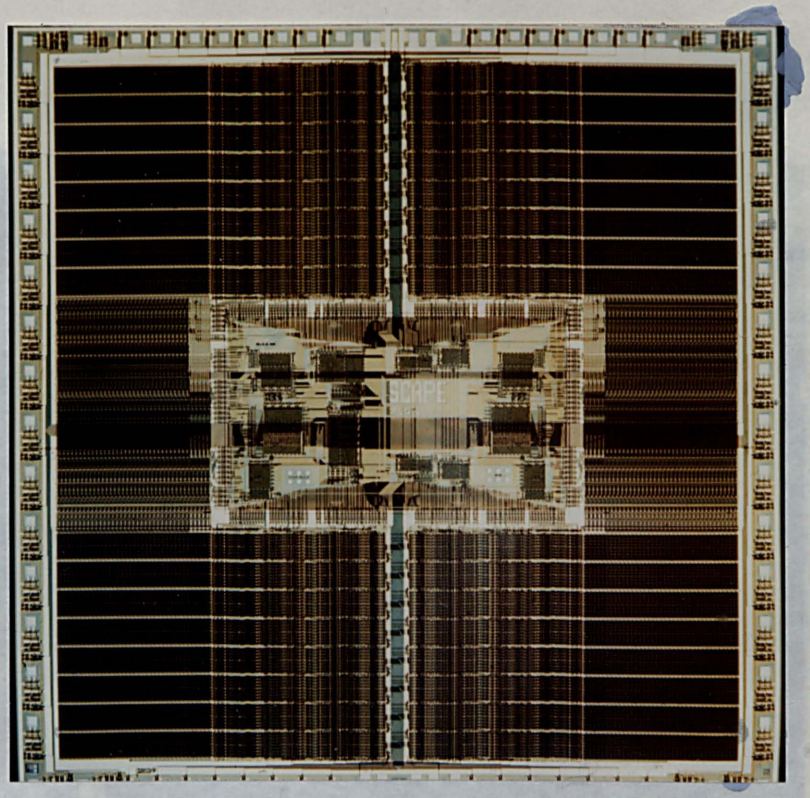


Figure 6-10. SCAPE chip microphotograph.

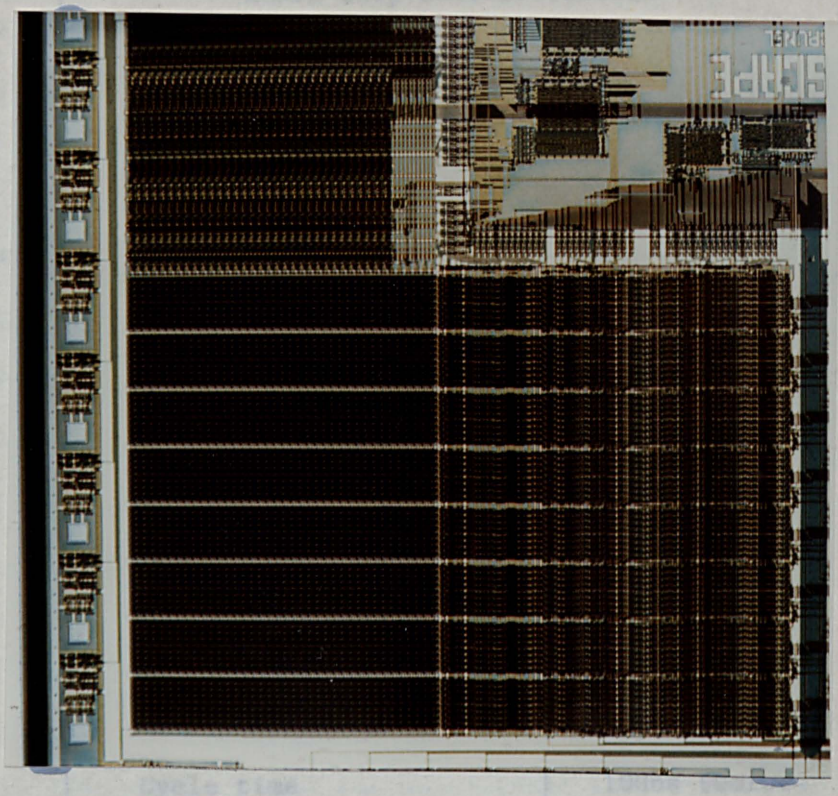


Figure 6-11. 64 x 37 CAM cell microphotograph.

The SCAPE chip has been fabricated (Figure 6-10.) and testing is in progress at the time of writing. To date, DC testing is complete and no faults have been found. Figure 6-11., is a microphotograph of a 64 x 37 SCAPE CAM array and its associated control logic. Table 6-14. gives performance characteristics of the SCAPE chip.

SCAPE chip characteristics	
SCAPE die size	75 sq.mm
transistor count	143,000
Memory array size	17.5 sq.mm
transistor count	76,000
BCL area	7.3 sq.mm
transistor count	13,200
WCL area	18.4 sq.mm
transistor count	41,500
Package	68-pin
I/O	TTL compatible
Memory organisation	(32+5) x 256
Cycle time	100ns (worst)
Power dissipation	865 mW

Table 6-14. SCAPE chip characteristics.

6.13 Conclusions

This chapter has reported the results of two experiments. Experiment 1 was intended to identify, and evaluate the electrical, physical and control characteristics of a range of associative memory designs.

To this end, a range of CAM designs have been simulated and their performance figures calculated. The layout design tradeoffs have been investigated. In particular, the CAM area overhead of the well separation has been exposed. In terms of performance, it is interesting to note that all the designs met the speed requirements. However, they had widely differing power requirements, with the nMOS designs having lower power consumption when toggling. The 'standard' CMOS design (CAM B) had particularly high power consumption both during write '0/1' and write 'X' operations.

In terms of control, it has been argued that resistive load CAMs (designs A, B and C), require large and powerful drivers. This has been borne out in the experience of the design of the control logic for the associative memory in the SCAPE chip. The capacitive load CAMs tend to have easier control overheads. In terms of power, the SCAPE chip design has shown that the CAM cell design tends to dominate overall chip power consumption.

Experiment 2 has evaluated the performance of these associative memory designs within the WASP environment. This experiment has clearly demonstrated that the operational environment has significant impact on the power characteristics of CAM designs. This strongly suggests that the selection of CAM designs should be determined not merely on 'raw' circuit performance, but by the behaviour of the circuit within the operational and architectural environment.

The results from this experiment suggest that at this stage CAM B with its relatively large layout area and high power consumption appears to be an unattractive choice for WASP implementation. CAM A has a very compact layout area but has a high power consumption and requires large control drivers. CAMs C, D and E have significantly

lower power consumption but involve a large area overhead. CAMs D and E have the added advantage of requiring simple control logic and drivers. Consequently, if silicon area is the over-riding constraint CAM A would appear best suited for WASP. If power constraints dominate then designs C, D or E may prove to be the better choice. If ease of controllability is also an important aspect then designs D or E may be preferred to design C.

The following chapter reports on Experiment 4, which aims to enhance the analysis of associative memories by investigating the yield and harvest implications that the use of the different associative memory designs has on WASP.

CHAPTER 7. YIELD IMPLICATIONS

7.1 Objectives

The objective of this chapter is to refine the evaluation of the CAM designs presented in the previous chapter by investigating the yield and harvest implications that the use of the different CAM designs has on WASP as discussed in Chapter 5, Section 5.3.3.

7.2 APE fault-tolerance

As has been already described (Chapter 4, Section 4.5) The WASP fault-tolerance scheme is a tree-based 'n out of m' strategy. For WASP, this implies obtaining 8192 working (and accessible) APEs out of the 12544 APEs integrated. In practice since some of the WI, CC, RAM and APP control logic may fail inspite of having incorporated extensive fault-tolerance, the target number of working (viz. defect-free) APEs will have to be somewhat higher than 8192.

Hedge and Lea [78] have investigated the design tradeoffs involved in implementing fault-tolerance at the APE level. Theoretically, the most yield-efficient approach would be to include every working APE into the associative string (Figure 7-1). In practice the overhead of including bypass logic in each APE proves cost-ineffective. Hedge and Lea demonstrate that by organising the APEs into sub-strings (viz. groups of APEs) and bypasssing faulty sub-strings (even if only 1 APE in the sub-string is faulty) results in a more silicon cost-effective fault-tolerance strategy. This mechanism is illustrated in Figure 7-2. where the sub-string bypassing is implemented in the 'row links'.

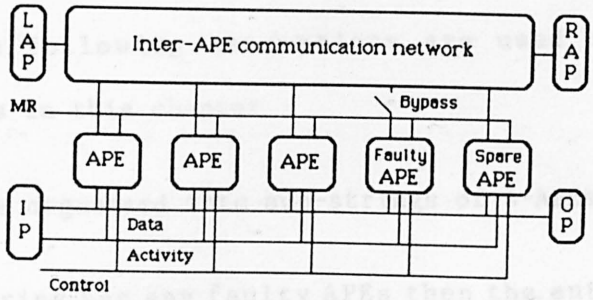


Figure 7-1. Ideal APE fault-tolerance mechansism.

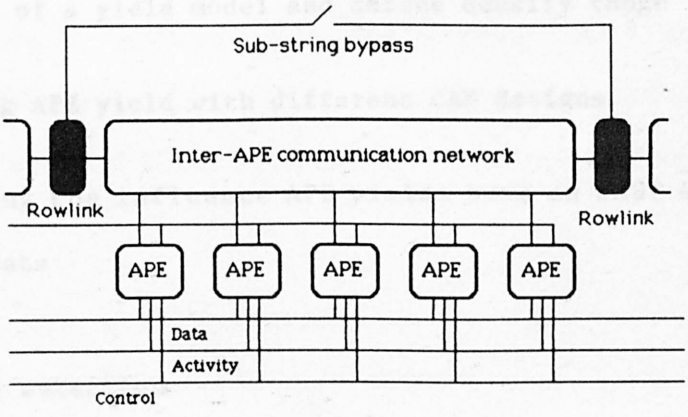


Figure 7-2. Practical APE fault-tolerance mechanism

In the same paper Hedge and Lea also investigate the design tradeoffs with different sub-string length for WASP. They determine that with the 64-APE WASP APP a sub-string length of 4 (viz. 4-APE 'blocks') provides the best tradeoff between yield and silicon overheads.

Accordingly, the following assumptions are used in the fault-tolerance analysis in this chapter

(1) All APEs are organised into sub-strings of 4-APEs

(2) If a sub-string has any faulty APEs then the entire substring is considered to be faulty.

(3) The overhead of implementing APE sub-string bypassing (predicted to be < 4% of APE area in [78]) is neglected.

7.3 Investigation strategy - experiment 3

The investigation strategy for this experiment comprises 3 stages, namely

(1) Selection of a yield model and defect density range

(2) Predicting APE yield with different CAM designs

(3) Estimating the influence APE yields have on WASP APE harvest requirements

7.4 Yield model selection

This section discusses the yield model and defect density used in the simulations.

7.4.1 Yield model

As the profitability of integrated circuit manufacture can be crucially dependent on the circuit yield, both academics and industrialists have attempted to develop accurate and sophisticated models for the prediction of chip yield.

One of the earliest and most widely-used models is based on a simple Poisson distribution where chip yield (Y) is approximated to

$$Y = e^{-da} \quad \ldots\ldots \quad (1)$$

where d is the average defect density (in defects per unit area) and a is the chip active area.

Many researchers (eg. Stapper, [79], Mangir [80]) have attempted to improve on the accuracy of this basic model, the increase in accuracy has largely been obtained by adding parameters to account for such phenomenon as clustering (viz. the propensity for one defect to 'attract' other defects to the same area). These parameters are specific to a particular process and have been largely determined by a 'best fit to data' approach [79]. Furthermore, as Stapper shows, these parameters vary widely from circuit to circuit, process to process, and even change if different process operatives are employed.

Consequently, for this experiment it is inappropriate to use 'sophisticated' models for the prediction of APE yield. Rather, the basic Poisson distribution (1) is adopted. Worries as to potential inaccuracies of this model have been allayed by the publication of Moore's work [81], who compares a wide range of yield models with the Poisson distribution and demonstrates that the basic Poisson model

provides a good approximation over a wide range of defect densities.

7.4.2 Defect density

Historically, the defect densities (viz. the average number of fatal fabrication failures per unit area) of IC processes have been closely guarded secrets. Indeed, the defect density of a particular process may itself change as process modifications are made. Often, once the defect density drops below a certain point, the feature size of the process may be changed (eg. a drop from 2.5 to 2 micron geometries) and defect densities will rise accordingly. Consequently it is necessary to use a range of defect densities in the yield modelling of the APEs.

A range of defect densities (0.02 to 0.1 d/mm^2 - 12.5 to 62.5 d/in^2) has been chosen for the yield modelling. This is slightly more wide-ranging than the (0.02 - 0.08 d/mm^2) spread suggested for most processes by Moore in [81].

7.5 Calculation of APE size

This section calculates the area of WASP APEs using different CAM designs.

7.5.1 Assumptions

The following assumptions based on the SCAPE chip have been made

(1) An APE comprises a 37-bit CAM word plus a WCL slice.

(2) The WCL slice area remains constant with different CAM designs and is assumed to be the same as in the SCAPE chip [8] (length

= 36uM and width = 2000uM).

7.5.2 APE areas.

Table 7-1. gives the CAM, 3-bit CAM word, WCL and APE areas for the 5 different CAM designs. The relative APE areas (using CAM A as a base of 1.00) are shown in Table 7-2.

CAM design	CAM area (uM^2)	37-bit word area (uM^2)	WCL area (uM^2)	1-APE area (uM^2)
A	1944	71928	72000	143928
B	2980	110260	72000	182260
C	4160	153920	72000	225920
D	5500	203500	72000	275500
E	5270	194990	72000	266990

Table 7-1. APE areas.

CAM design	Relative CAM area	Relative APE area
A	1.00	1.00
B	1.53	1.27
C	2.14	1.56
D	2.82	1.91
E	2.69	1.84

Table 7-2. Relative CAM and APE areas.

From these tables it can be seen that using the larger CAM designs incurs a noticeable area overhead (eg. APEs that use CAM B are 1.27 times larger than an APE using CAM A). However, the fact that an APE comprises a CAM word plus associated control logic means that the increase in APE area does not scale with the increase in CAM area (eg. even though CAM D is 2.82 times as large as CAM A, an APE using CAM D is only 1.91 times larger than an APE using CAM A).

7.6 APE yield and harvest requirements

This section investigates the APE yield and WASP harvest requirements using the different CAM designs.

7.6.1 WSI metrics

The following set of WSI metrics developed from those introduced by Jones and Lea [64] are used to measure the APE yield and harvest.

(1) Available storage (AS), this measures the average number of APEs expected to be working on the wafer and is defined as

$$AS = Y(APE) \times N(wafer)$$

where Y(APE) is the probability of a single APE being functional and N(wafer) is the number of APEs on the wafer

(2) Harvest (H), measures the proportion of working APEs that need to be utilised to achieve the desired number of useable APEs. Harvest is defined as

$$H = DS / AS$$

where AS is as defined above and DS is the desired storage

(viz. 8192 APEs in the WASP image processing device)

7.6.2 Results - experiment 3

The results section is split into 2 sections: firstly the APE yield results and secondly the harvest implications.

7.6.2.1 APE yield

Table 7-3. gives the Y(APE) figures for WASP APEs using the 5 different CAM designs. Table 7-3. Is based on the following assumptions

(1) As the APE sub-string length is 4-APEs Y(APE) is based on the yield of a 4-APE block

(2) Poisson statistics used for yield modelling ($Y = e^{-da}$)

CAM cell	4-APE area	Defect density (d/mm^2)				
	(mm^2)	0.02	0.04	0.06	0.08	0.10
A	0.57571	98.86	97.72	96.60	95.50	94.41
B	0.72904	98.55	97.13	95.72	94.33	92.97
C	0.90368	98.21	96.45	94.72	93.03	91.36
D	1.10200	97.82	95.69	93.60	91.56	89.57
E	1.06796	97.89	95.82	93.80	91.81	89.87

Table 7-3. Y(APE) (%) for a 4-APE substring.

From Table 7-3. it can be seen that the 4-APE blocks are quite compact and that this gives them all a relatively high yield. Indeed, despite the largest APE (using CAM D) being nearly twice the size of

the smallest APE (using CAM A) it still is possible to achieve an 4-APE yield of nearly 90% with the worst-case defect density. At the more modest defect densities (eg. 0.02-0.04 d/mm^2) the difference in yield between the 2 extremes of APE size is less than 2%.

7.6.2.2 Functional APEs per wafer

The AS (viz. the average number of functional APEs expected per wafer) is given in Table 7-4. and is based on the following assumptions

(1) The WASP IP device has 196 APP modules

(2) Each APP module comprises 64 APEs organised as one string of 16, 4-APE sub-strings

(3) Defect density is uniform across the wafer

On these assumptions, AS can then be calculated as

$$AS = Y(APE) \times N(wafer)$$

with Y(APE) as given in Table 7-3 and N(wafer) = (196 x 64) = 12544.

The AS for WASP using different APEs is given in Table 7-4 for the same range of defect densities. These figures represent the average number of APEs we can expect to be working on a WASP IP wafer. From this Table 2 main points emerge, namely

(1) The difference in AS figures for the different CAM designs is quite small. Even at the highest defect densities considered, there is a difference of only 607 APEs between CAM D (the largest APE option) and CAM A (the smallest APE option).

(2) All APE options are highly silicon-efficient. APE yields are never below 89% even with the largest CAMs and highest defect density. At a more modest defect density (eg. 0.04 d/mm^2) every option yields over 95% functional APEs.

CAM cell	4-APE area	Defect density (d/mm^2)				
	(mm^2)	0.02	0.04	0.06	0.08	0.10
A	0.57571	12400	12257	12117	11979	11842
B	0.72904	12362	12183	12007	11832	11662
C	0.90368	12319	12098	11881	11669	11460
D	1.10200	12270	12003	11741	11485	11235
E	1.06796	12279	12019	11766	11516	11273

Table 7-4. AS for WASP wafer (4-APE sub-string).

These encouraging figures are due to the granularity of the WASP fault-tolerant strategy. Since the unit of bypass is a 4-APE sub-string. In practice, using any of the 5 CAM cell designs the area of a 4-APE sub-string still remains very small. Consequently, the yield of all the 4-APE sub-string options is very high. Thus in every case, large number of APEs can be expected to be functional.

7.6.2.3 Harvest requirements

The harvest requirements of the AS figures in Table 7-4 (4-APE sub-string) can be calculated by using the H metric in section 7.6.1. where H (the proportion of working APEs that need to be harvested is given by

$$H = DS / AS$$

where DS for the image processing WASP device is 8192 and AS is given in Table 7-4.

Applying the harvest metric gives the data in Table 7-5.

CAM cell	4-APE area	Defect density (d/mm^2)				
	(mm^2)	0.02	0.04	0.06	0.08	0.10
A	0.57571	66.06	66.84	67.61	68.39	69.18
B	0.72904	66.27	67.24	68.23	69.24	70.25
C	0.90368	66.50	67.71	68.95	70.20	71.48
D	1.10200	66.76	68.25	69.77	71.33	72.91
E	1.06796	66.71	68.16	69.62	71.13	72.67

Table 7-5. Harvest requirements (%) for WASP wafer (4-APE sub-string).

From this table it can be seen that the harvest requirements for WASP using the different CAM designs vary approximately in the range 66-73%. Naively, this can be thought of as 66-73% of the WI-CC-RAM-APP controller paths must be functional, if 8192 APE WASP IPs are to be frequently harvested.

It is interesting to note that due to the fine-granularity of the APE fault-tolerance, the APE harvest requirements change remarkably little over a 5-fold change in defect density.

7.7 Implications

This section discusses the implication that the use of different CAM designs has on WASP yield and harvest.

Table 7-6. summarises the relative yield implications for the 5 different APE options using the smallest APE option (CAM A) as a base and assuming the worst case defect density (0.1 d/mm^2).

Name	Relative APE area	Relative AS
A	1.00	1.00
B	1.27	0.98
C	1.56	0.97
D	1.91	0.95
E	1.84	0.95

Table 7-6. Relative WASP yield data (4-APE sub-string) and 0.1 d/mm^2.

This data strikingly demonstrates the advantages of the fine-grain ´n out of m´ APE fault-tolerance strategy. The efficiency of this approach of utilising all 4-APE sub-strings means that while the use of different CAM designs results in the largest APE option (CAM D) being nearly twice the area of the smallest APE option (CAM A), in yield terms it means that the largest APE option (CAM D) still provides 95% as many APEs as the smallest APE option (CAM D). Thus the fault-tolerant strategy of WASP means that while the use of different CAM designs noticeably alters APE area it has a relatively insignificant effect on APE yield. Consequently, in the evaluation of CAM designs for WSI provided a fine-granularity ´n out of m´ approach is used, the yield implications of CAM design does not appear to be a major problem.

7.8 A CAM test chip

Collaboration with an Alvey project in WSI (VLSI/ARCH/073) resulted in an opportunity to fabricate the CAM designs as part of a combined Brunel/Plessey test chip. This chip could be used to verify the yield analysis and provide further data on CAM failure modes as well as confirming the CAM performance figures of Chapter 6.

The floorplan of the chip (Figure 7-3.) comprises 4 seperate 8-bit x 12-word CAM arrays of designs B, C, D and E, (Figures 7-4, 7-5, 7-6 and 7-7). with IO multiplexing to reduce pin-out. A version of design A had been fabricated on SCAPE. Each quadrant has simple uBCL (micro bit-control logic) and uWCL (micro word-control logic) to support associative memory manipulation. Table 7-7. summarises the CAM test chip characteristics.

CAM test chip data	
Area	12.5mm^2
Pin out	48
# CAM designs	4
CAM array size	8-bits x 12-words
# CAM cells	384 (4 x 8 x 12)
Transistors	6000 (approx)

Table 7-7. CAM test chip details

The chip has been fabricated (Figure 7-8.). However, the Alvey collaborators wished to fully test their portions before handing the wafers over to Brunel. Frustratingly, nearly a year after the design had been submitted for fabrication, the wafers have not been transfered to Brunel for testing.

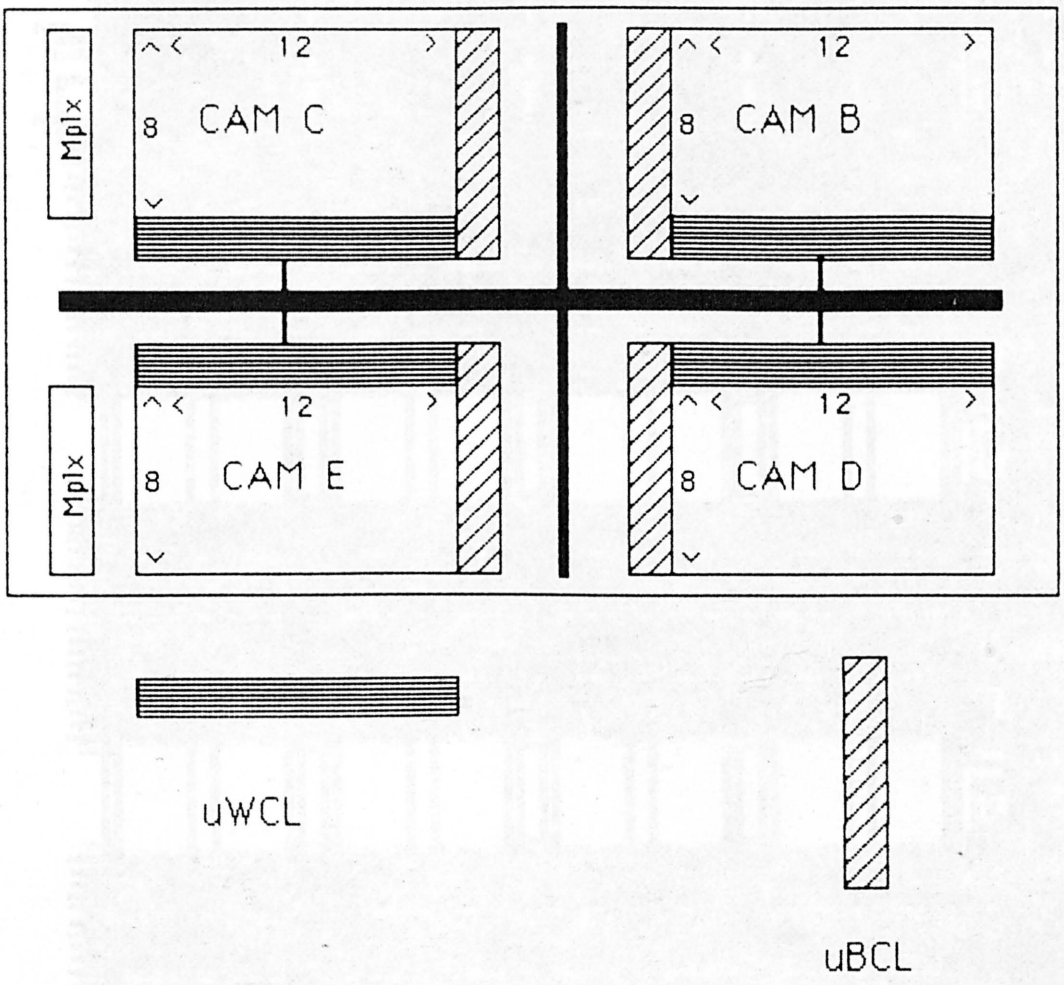


Figure 7-3. CAM test chip floorplan.

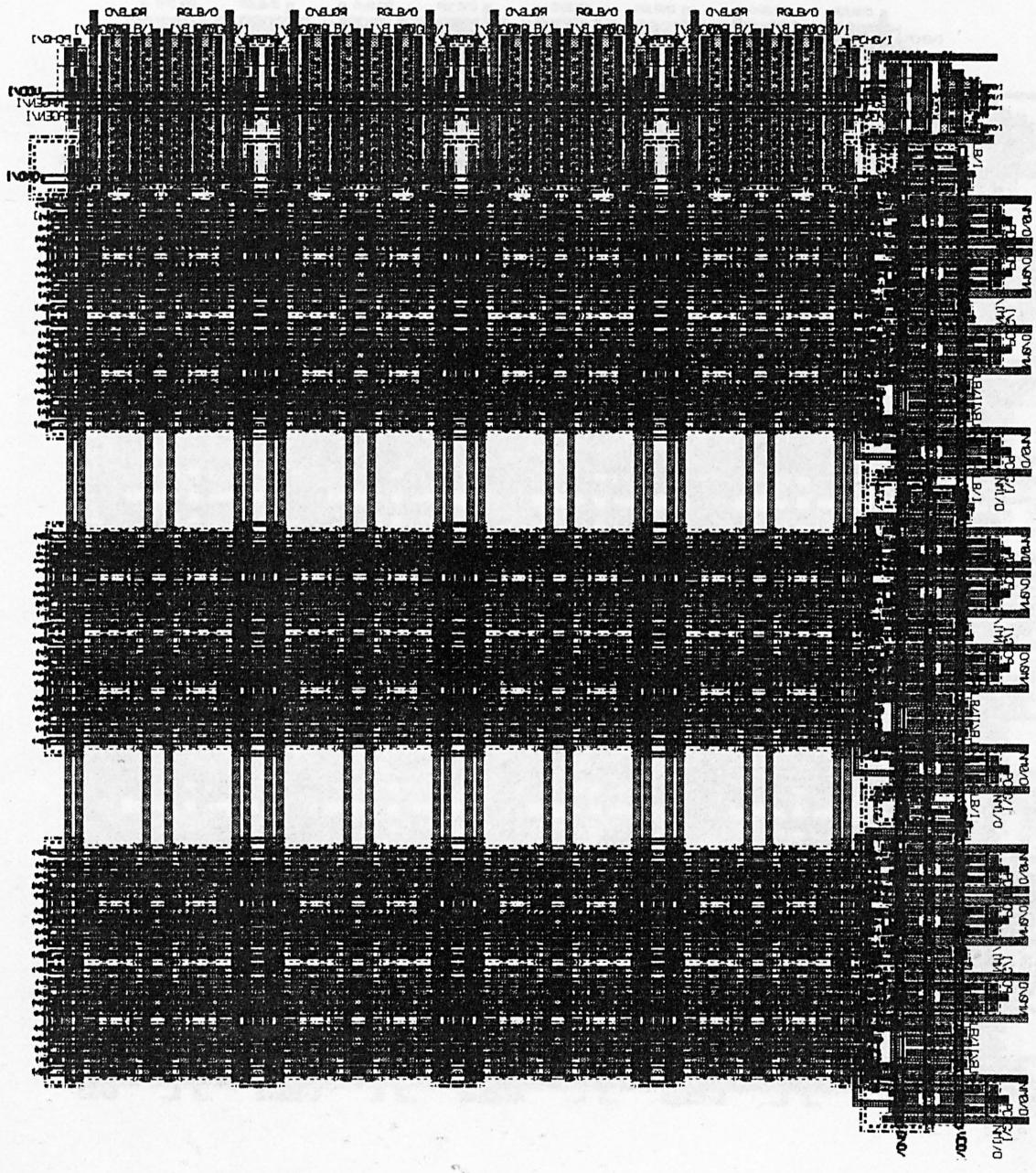


Figure 7-4. CAM B array.

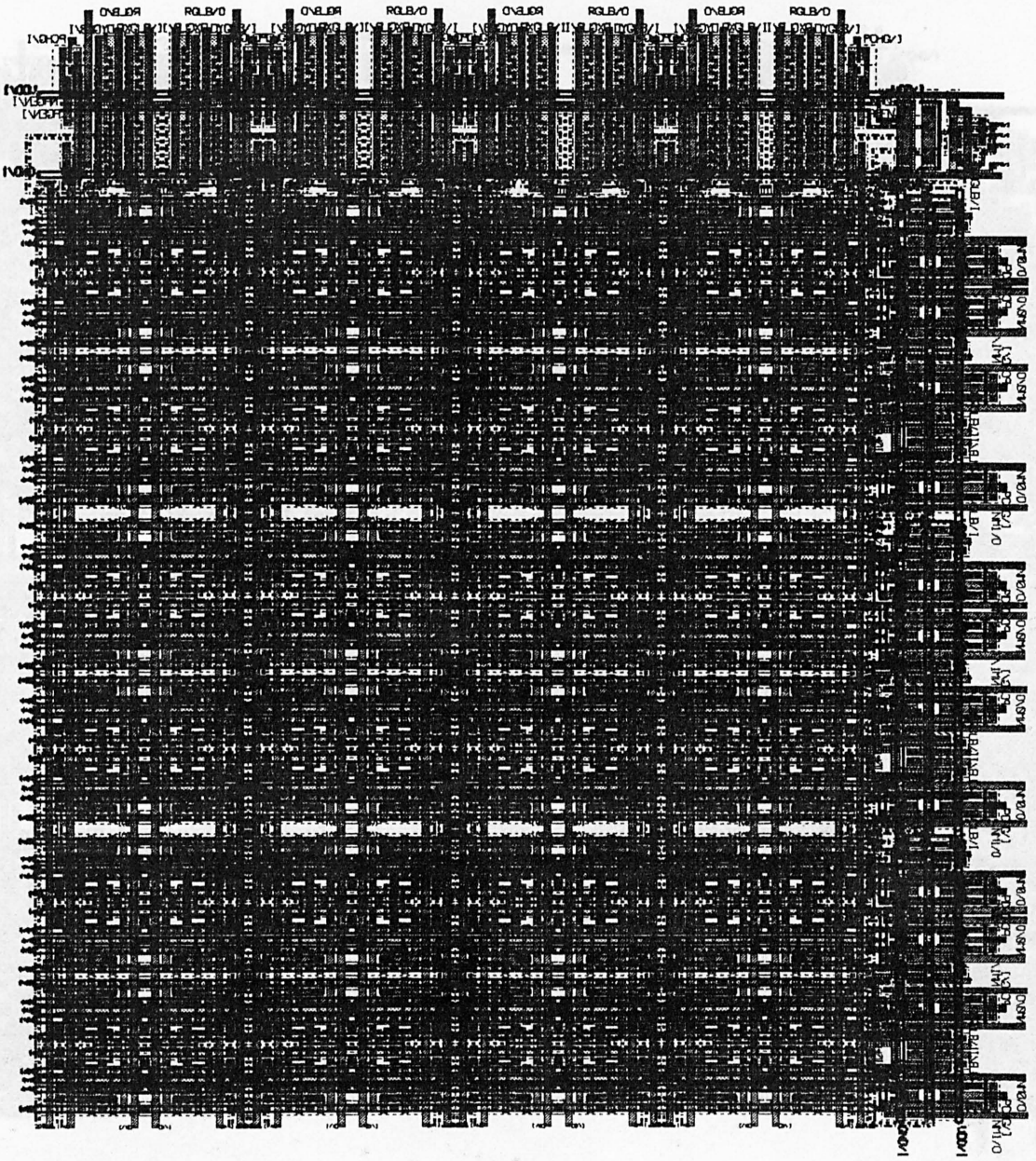


Figure 7-5. CAM C array.

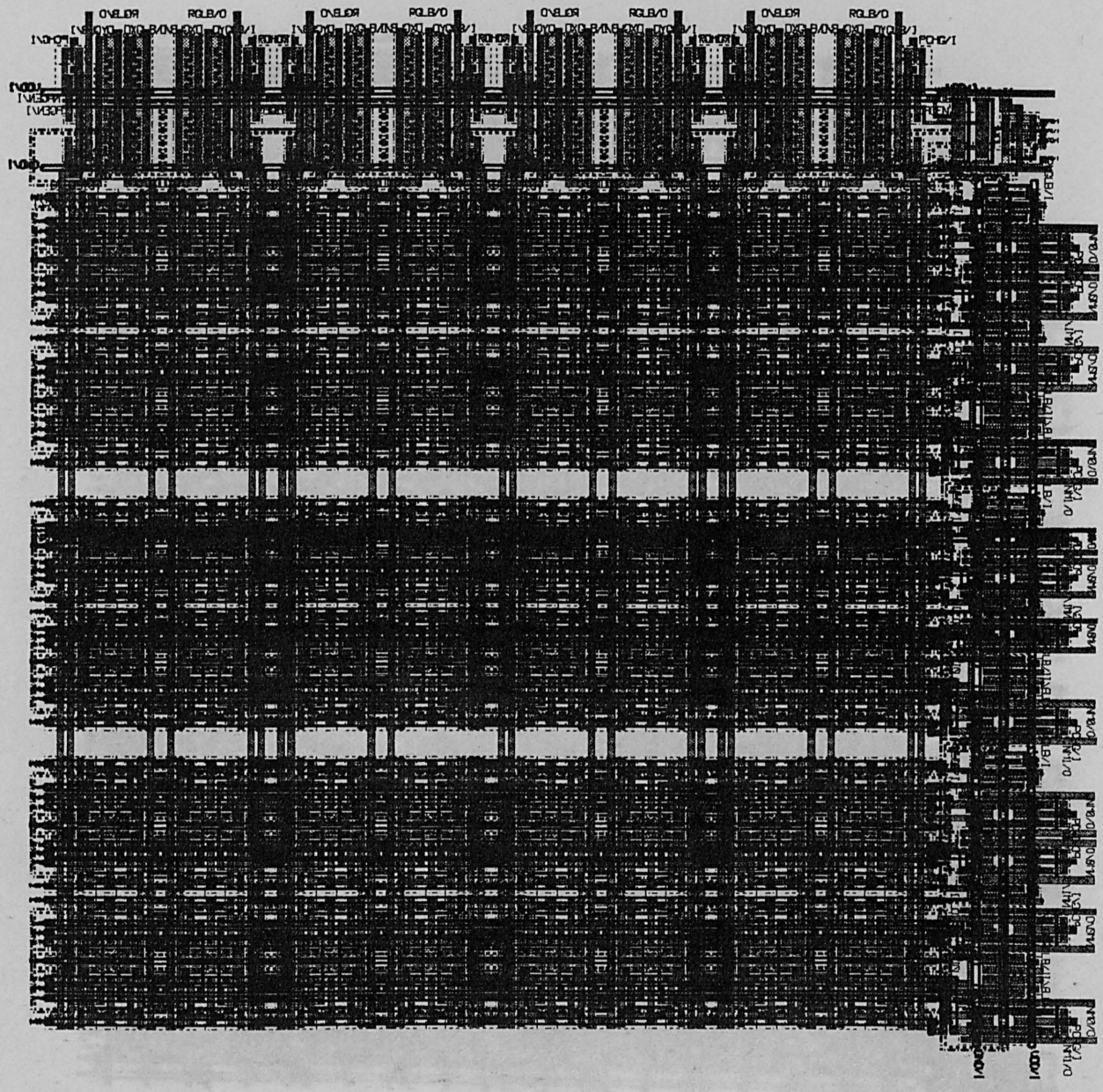


Figure 7-6. CAM D array.

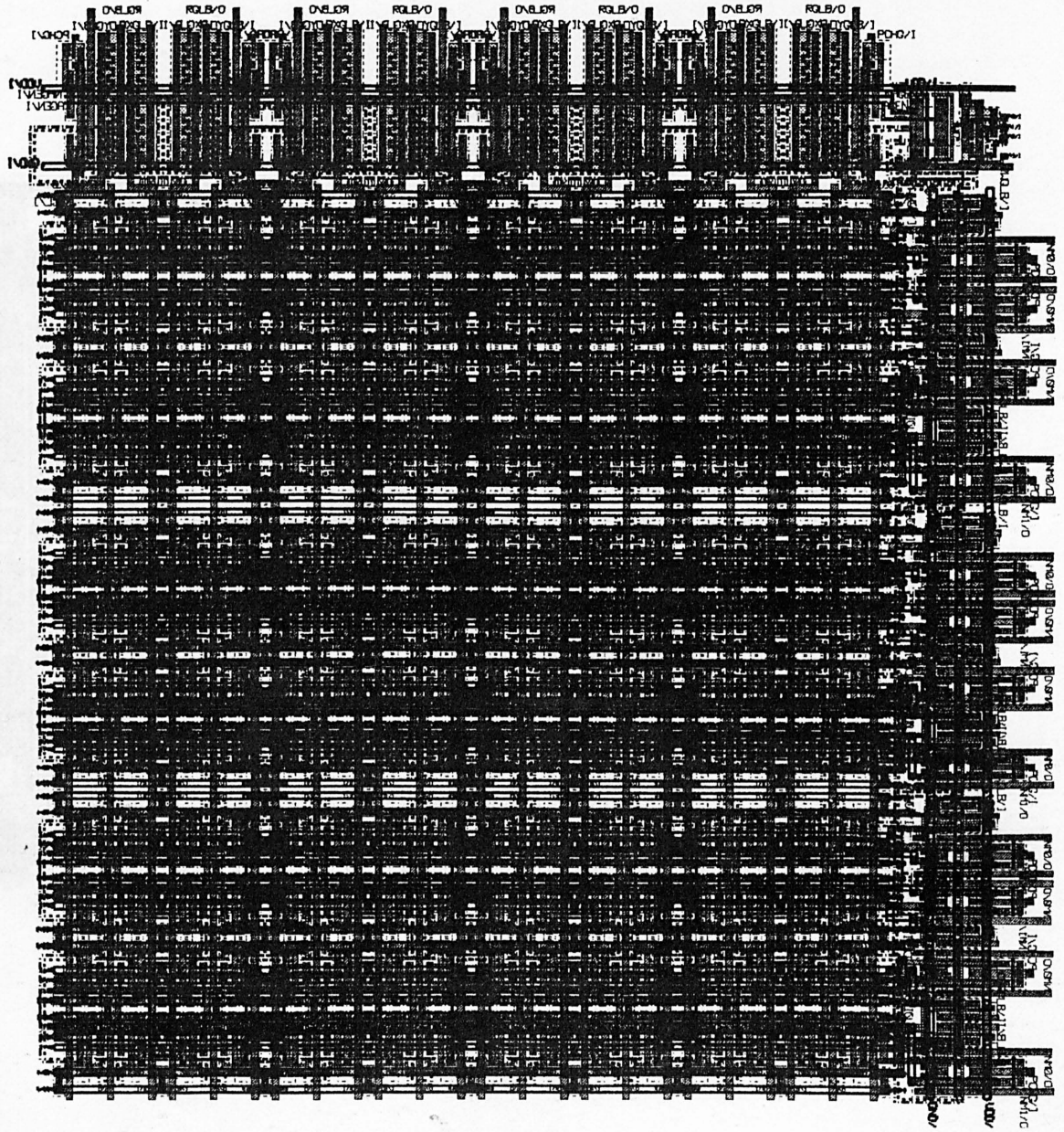


Figure 7-7. CAM E array.

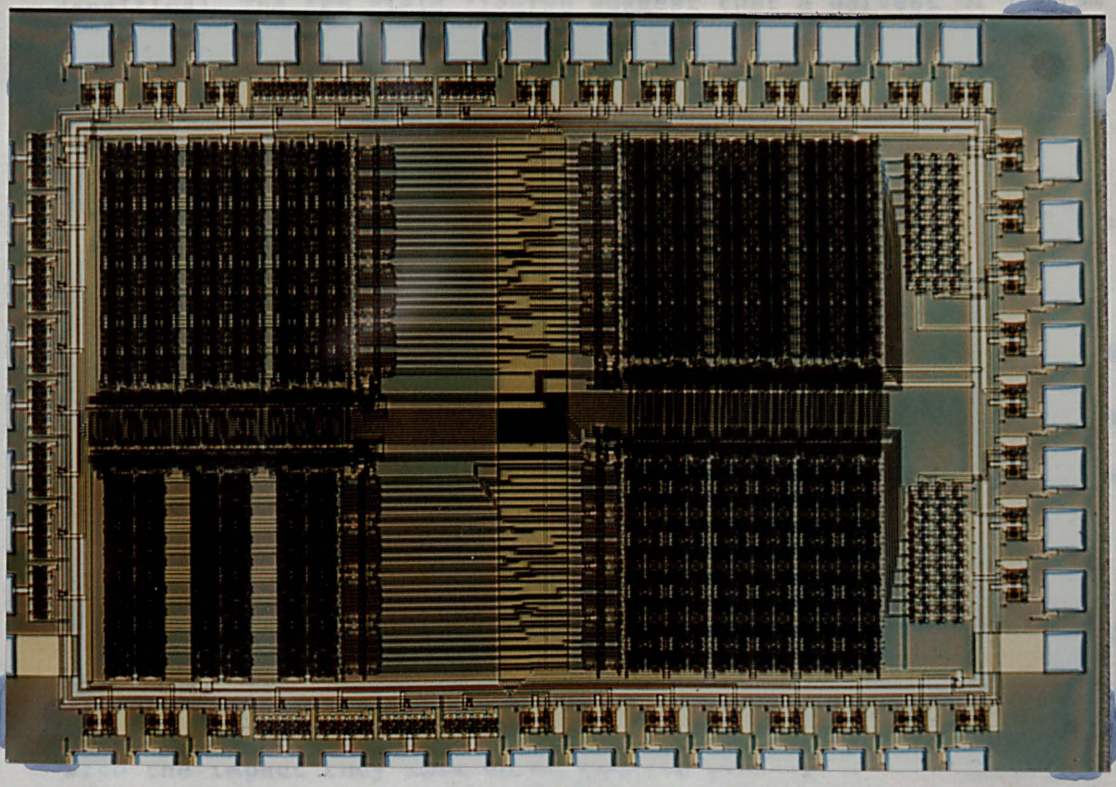


Figure 7-8. CAM test chip microphotograph.

7.9 Conclusions

This chapter has investigated the yield implications of using the different CAM designs in WASP. A set of metrics for the evaluation of APE yield have been suggested. Use of these metrics demonstrate that the fine-granularity 'n out of m' approach of the WASP APE fault-tolerance strategy is highly effective. Indeed, it is possible to increase the APE size almost 2-fold and see only a slight decrease in the number of available APEs even at the more severe defect densities). The APE yield figures suggest that a harvest of 65-75% is sufficient to allow WASP to achieve the desired 50% 8192-APE wafer target.

Overall the results in this chapter suggest that if a fine-grain 'n out of m' fault-tolerant strategy is used, its yield insensitivity to relatively large changes in PE area means that all the CAM designs easily meet the yield and harvest requirements. Perhaps surprisingly then, they all are virtually equally acceptable for use in WSI from a yield and harvest standpoint.

The following chapter investigates how well the associative memory designs meet the electrical and physical constraints of WSI, together with the impact they have on PE packing density.

CHAPTER 8. IMPLEMENTATION IN WASP

8.1 Objectives

The objectives of this Chapter concern Experiments 4 and 5 described in Chapter 4, namely

(1) To determine the suitability of the range of associative memories previously described for implementation in WSI (Experiment 4).

(2) Evaluate the influences different associative memory designs have on WASP PE packing density (Experiment 5).

8.2 Investigation strategy - experiment 4

Chapter 4 has shown that WASP has a potentially effective and efficient fault-tolerance strategy. Chapter 6 has demonstrated that all 5 associative memory designs meet the speed requirements of WASP, but that their power characteristics vary dramatically. Consequently, the relevant WSI implementation issues investigated are electrically and physically oriented and concerned with the supply of power to the wafer and the removal of the heat generated.

The first part of the investigation involves the identification of the power supply and power dissipation constraints in WSI. Having achieved this, the next stage is to evaluate the power density for WASP APPs using different CAM cell designs. This can then be used to estimate the power required per WASP wafer and hence to evaluate how well the different associative memory designs meet the electrical and physical constraints of WSI.

8.3 Evaluation of constraints

The 2 main constraints are thermal management and power supply distribution.

8.3.1 Thermal management

In [82,83], McKirdy and Lea look at physical design issues in WSI. Their investigation into thermal dissipation limits (for an 80°C junction rise) is summarised in Figure 8-1. These figures suggest that wafer power dissipation poses a major problem, as they are about an order of magnitude lower than for the corresponding VLSI chips. To place these power density figures in context, Table 8-1. relates these power density figures to wafer power dissipation limits for 4, 5 and 6 inch wafer sizes.

Cooling technique	Max dissipation (W) Wafer size (inches).		
	4	5	6
Natural convection	11	16	23
Forced air (4m/s)	36	50	66
Fast forced air (8m/s)	51	71	94

Table 8-1. Power dissipation limits in WSI (from [79])

80°C junction rise.

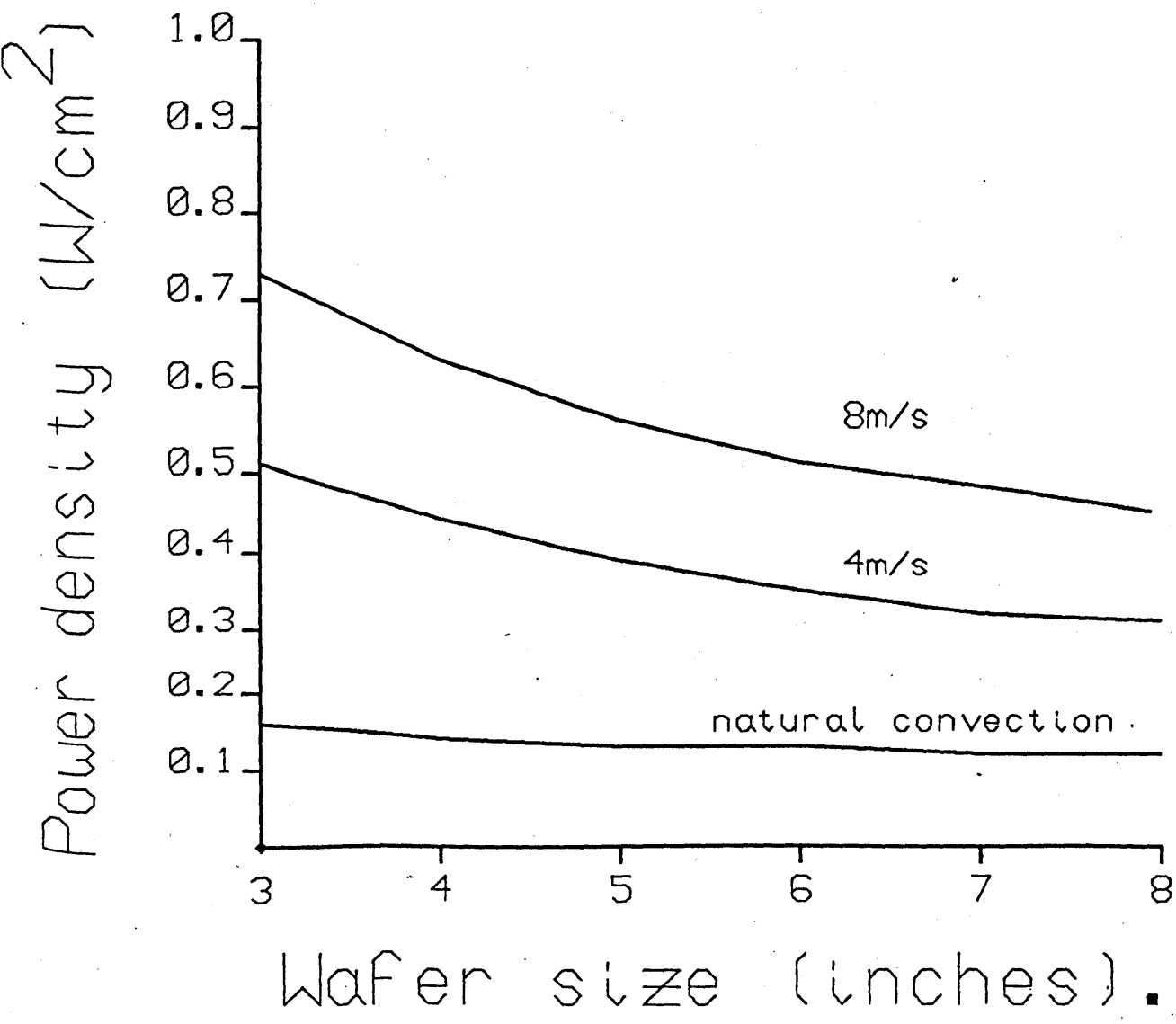


Figure 8-1. Thermal dissipation limits in WSI.

A 4 inch wafer using natural convection cooling is restricted to dissipating less than 11 W. Using a small fan capable of forcing air over the wafer at 4m/s (as found in many microcomputer systems) allows 36 W to be dissipated. Even using a large and powerful 8m/s fan (as found in many air-cooled mainframes) still restricts the wafer (for example, comprising in WASP, a complete patch processing module) to just over 50 W. This severe limit in power density is a striking demonstration of the consequences of the much increased packaging density achievable in WSI.

8.3.2 Power distribution network

In [74], it is demonstrated that an independent rails power distribution strategy (Figure 3-7.) appears to be attractive to WSI implementation because,

(1) The structure is fault-tolerant since power distribution is divided amongst several networks and a failure in one network only affects a portion of the wafer.

(2) Only one layer of metal is required, allowing other layers to be freely utilised for control and data distribution.

(3) It is amenable to low cost fabrication, using conventional 'step-and-repeat' technology as the physical positioning of the rails in each reticle is identical.

Consequently, an independent rails power distribution strategy has been selected for the WASP device. However, the independent rails strategy, like all WSI power distribution strategies requires large power distribution tracks as,

(1) The system power distribution network has moved from the board onto the silicon device itself. Thin high-resistance Aluminium tracks now have to perform the same function as thick low-resistance Copper tracks on a PCB.

(2) Modules in WSI are responsible for the supply of power to neighbouring modules, resulting in a module power distribution overhead potentially many times greater than that required for a single module.

(3) Tracks in WSI systems may run for distances approaching 10 cm with a correspondingly high resistance, whereas in VLSI the longest power track is often no more than 0.5 cm.

These factors suggest strongly that the power distribution area overhead in WSI may be significantly larger than in VLSI.

Silicon area occupied by power distribution tracks reduces the area available for 'productive' communication. Typically, 2 layers of metal are needed to route signals (for example, the SCAPE chip orthogonally routes data in 1st layer metal and control and power in 2nd layer metal) and if a large amount of one layer of metal is required for power distribution, it may significantly reduce the area available for routing signals. Furthermore, failures in the power network could make major units of the wafer inoperable. The larger the power network area, the larger the target area it presents. Consequently, an early identification of the area of the power distribution network is an important engineering objective in the design of a WSI-based system.

In order to more accurately forecast the impact of power density on power network distribution area, the following section derives from

first principles, an equation relating power density and WSI power track area.

8.3.3 Derivation of power network area formulae

A power rail supplying n modules can be modelled as a string of resistors, hence the voltage drop can be expressed as:

$$V = I_1R_1 + I_2R_2 + \ldots\ldots I_nR_n$$

But if the power rail is of uniform dimensions,

$$R_1 = R_2 = R_n = R$$

hence,

$$V = R(I_1 + I_2 + I_n)$$

For uniform power density across the wafer,

$$I_1 = nI_{mod},\ I_2 = (n-1)I_{mod},\ \ldots\ldots I_n = I_{mod}$$

Where I_{mod} is the average current density per module.

Hence,

$$\sum I = \frac{I_{mod}\ n(n+1)}{2}$$

Therefore,

$$V = \frac{R\ I_{mod}\ n(n+1)}{2}$$

$$V = \frac{R_{\#}\ L_{mod}\ I_{mod}\ n(n+1)}{2W}$$

Where L_{mod} = length of module

W = width of power rail

$R_{\#}$ = is the unit square resistance of the power track

Rearranging we get,

$$W = \frac{R_{\#} L_{mod} I_{mod} n(n+1)}{2V}$$

The proportional area (Ap) of the module occupied by 2 power rails is equivalent to

$$Ap = \frac{2W}{W_{mod}}$$

Where W_{mod} is the width of the module

Giving,

$$Ap = \frac{R_{\#} L_{mod} I_{mod} n(n+1)}{V W_{mod}}$$

V represents the voltage drop on 1 rail. A more useful measure would be the total voltage drop (VD). If VD represents the total voltage drop (viz. $(V_{dd} - V_{dd}')$ + $(V_{ss}' - V_{ss})$ where V_{dd} and V_{ss} are the power and ground voltages at the wafer pins and V_{dd}' and V_{ss}' are the voltage levels of power and ground at the centre of the wafer) then the relationship between V and VD is

$$V = \frac{VD}{2}$$

Hence,

$$Ap = \frac{2R_{\#}\ L_{mod}\ I_{mod}\ n(n+1)}{VD\ W_{mod}} \quad \ldots\ldots\ldots (1)$$

Seperating equation (1) out, it can be seen that

$$\frac{2R_{\#}}{VD}$$

is usually process defined (as VD is severely restricted by the design rules, often to < 0.5V to avoid latch-up in CMOS).

$$\frac{L_{mod}}{W_{mod}}$$

is determined by layout constraints and in the WASP device is unity (eg. $L_{mod} = W_{mod}$).

$$I_{mod} \text{ and } n$$

are determined by the architecture and wafer size respectively.

Assuming,

(1) process specific factor $\frac{2R_{\#}}{VD}$ = 160 milliohms/V, (ie. $R_{\#}$ = 40 milliohms and VD = 0.5V) This is typical of a modern CMOS process where V_{dd} must not drop below 4.75V nor V_{ss} rise above 0.25V and standard Aluminium tracks are used to supply power.

(2) $L_{mod} = W_{mod}$ = 1cm

Then,

$$I_{mod} = \frac{P_{mod}}{(V_{dd} - V_{ss})} \quad \ldots\ldots\ldots (2)$$

where P_{mod} is the module power in W and I_{mod} is the module current in Amps. Furthermore, since in assumption (1) VD is

0.5V then $(V_{dd}' - V_{ss}')$ = can be considered to be 4.5V

Since the module area is assumed to be in cm^2, then

$$P_{mod} = PD_{mod}$$

where, PD_{mod} = module power density in W/cm^2.

(3) To convert the number of modules (each 1cm on a side) to wafer diameter in inches, taking the square within the circle and using Pythagoras's theorem,

$$(2.54D)^2 = 2X^2$$

where X is the distance along the edge of the wafer.

$$2.54D = 1.41X$$

As X is the distance along one edge of the wafer square, then the relationship between the distance into the centre of the wafer (n) and X is

$$X = 2n$$

therefore,

$$2.54D = 2.82n$$

where D is the wafer size in inches

hence,

$$n = \frac{2.54D}{2.82} = 0.90D \quad \ldots\ldots\ldots \quad (3)$$

then, substituting (2) and (3) in (1) and as previously stated 4.5V for $(V_{dd}' - V_{ss}')$ gives

$$Ap = \frac{0.16\ PD_{mod}\ 0.90\ D(D+1)}{4.5}$$

which simplifies to

$$Ap = 0.032\ PD_{mod}\ D(0.90D+1) \quad \ldots\ldots\ldots \quad (4)$$

Plotting Ap against D for a range of power densities using (4) results in Figure 8-2. It can be seen that even with only modest power densities, the area occupied by the power distribution strategy can be alarmingly high with contemporary wafer sizes, and only ultra-low power densities ($< 0.1\ W/cm^2$) appear cost-effective with 8 inch slices. Furthermore, architectures that exceed the 100% overhead (eg. a $1\ W/cm^2$ architecture on a 5 inch wafer) are infeasible even if an extra layer of metal is added to the process. Moreover, process improvements that result in reduced metal resistivity (equivalent in Figure 8-2. to a reduction in power density) need to offer at least an order of magnitude improvement before VLSI power densities become feasible in WSI from a power supply viewpoint.

8.3.4 Comparison of limits

This section demonstrates the effect that these limits have on different wafer sizes.

The following assumptions have been made

(1) The maximum air flow speed for heat removal is 8 m/s.

(2) A 'reasonable' criteria as to the proportion of wafer area that is allocated to routing power tracks is 20% (ie. Ap = 0.2). By way of comparison, the power rails in the VLSI chip SCAPE [8], occupy 14.1% of total chip area (ie. Ap = 0.141).

Figure 8-3. relates the maximum thermal dissipation (based on McKirdy and Lea's data) for a range of wafers using natural convection cooling, together with the maximum power density that can be supported assuming 20% power rail area.

It can be seen that the thermal dissipation power density remains relatively constant. However, the power density that can be supported by the power supply rails drops rapidly with increasing wafer size. Nonetheless, for all but the largest wafer sizes, the lower limit is the thermal dissipation limit. Consequently, using natural convection cooling, the architecture (based on the 20% power rail area assumption) is mainly limited by thermal dissipation. This also suggests that using natural convection cooling, less than 20% power rail area is required.

This limit is very low. Indeed, it is about an order of magnitude below that of current VLSI chips, and in an effort to increase the power density limit a small (eg. 4 m/s air flow) fan could be employed.

Figure 8-4., relates power density for a range of wafers using 4 m/s forced air cooling, to the power density that can be supported with a 20% power rail area allocation. The situation is almost completely reversed. For all but the smallest wafers, the lower limit is that defined by the power rail overhead. This suggests that using a more powerful fan (eg. 8 m/s air flow) offers little improvement in WSI power density due to the limitation of power rail area. This is borne out by Figure 8-5., where for all wafers larger than 3 inches (assuming a limit of 20% power track area) power density limits are the same as for a 4 m/s fan.

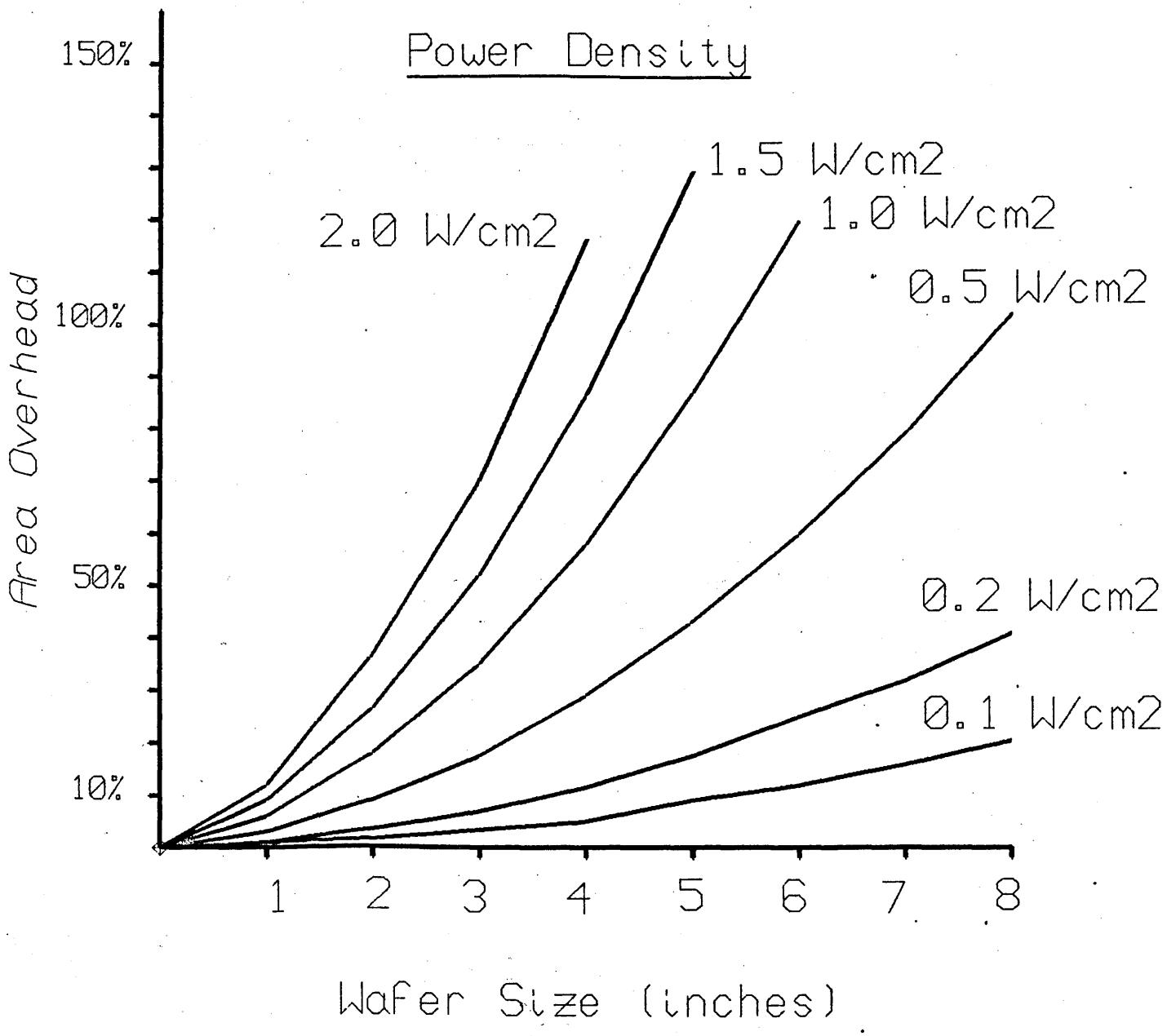


Figure 8-2. Power distribution limits in WSI.

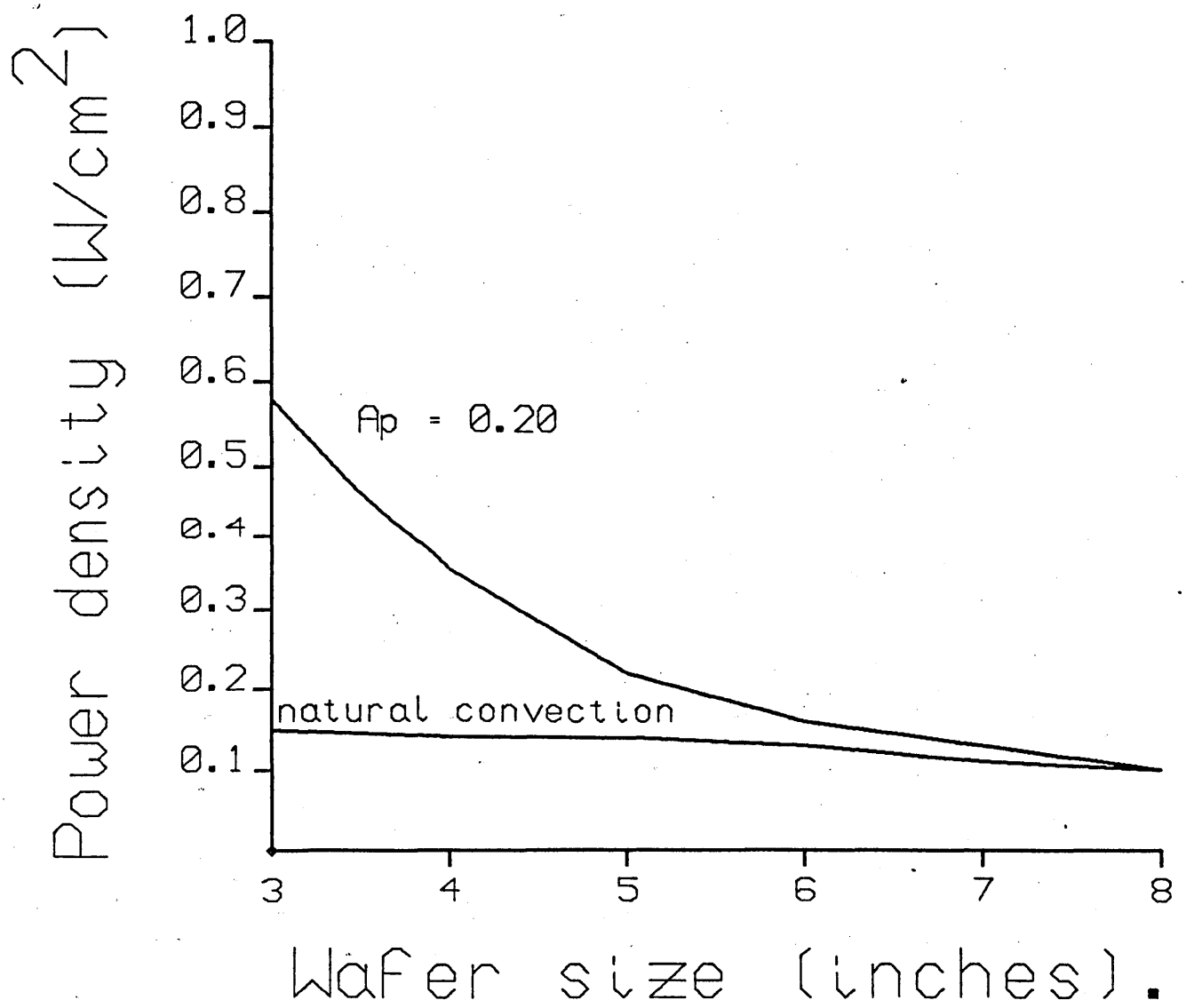


Figure 8-3. Natural convection,

20% power distribution area.

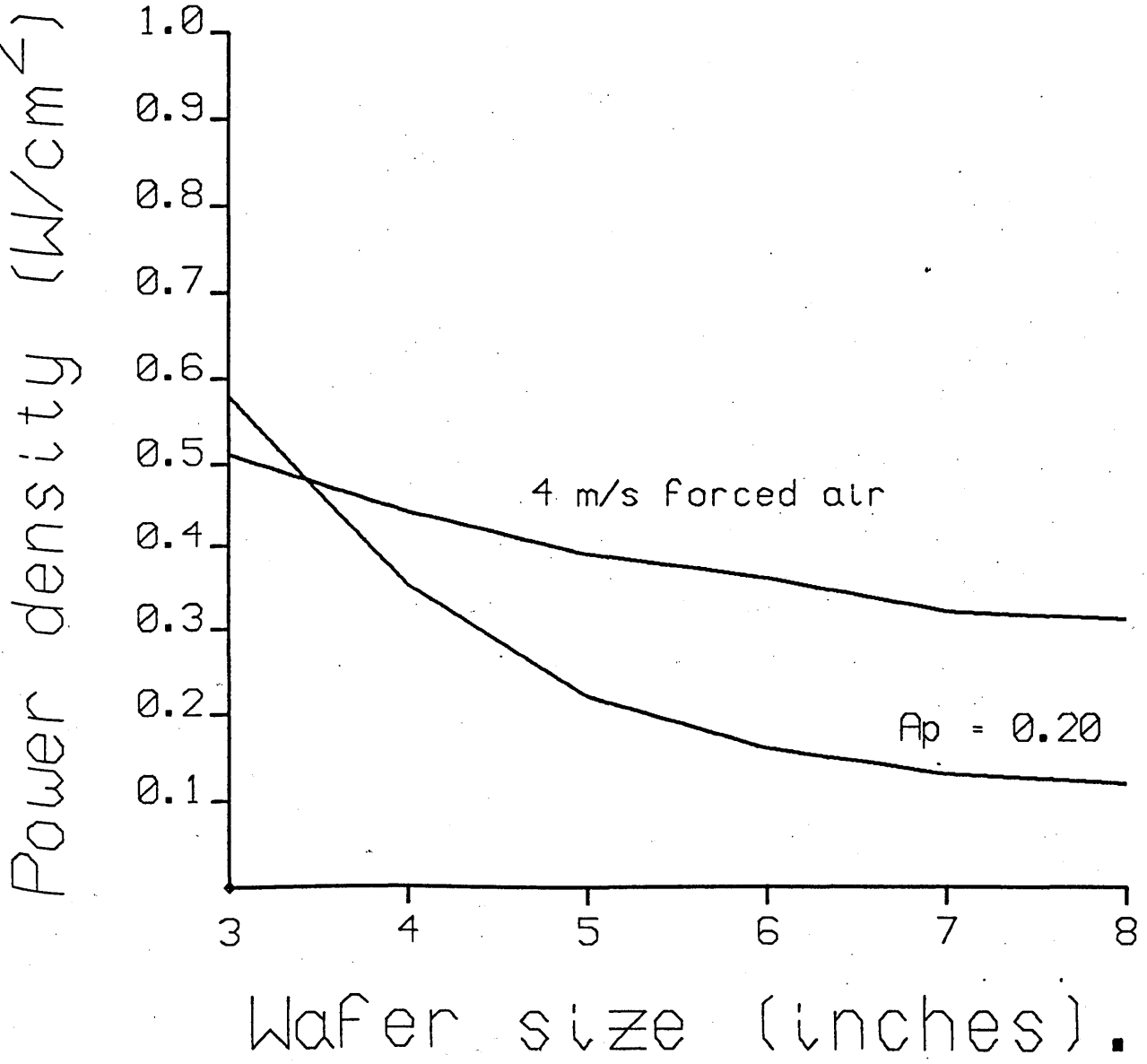


Figure 8-4. 4 m/s Forced air,

20% power distribution area.

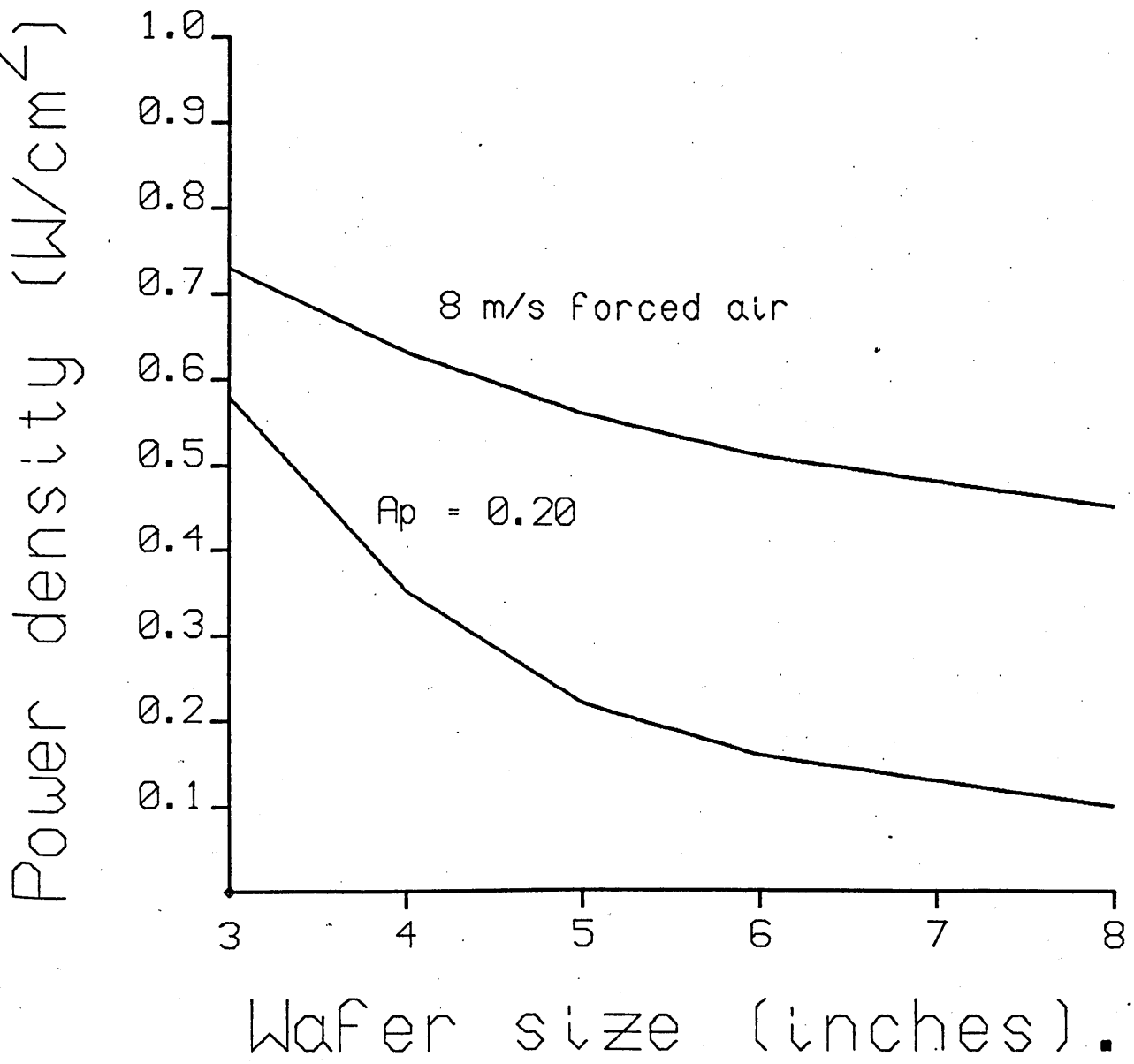


Figure 8-5. 8 m/s forced air,

20% power distribution area.

The low-cost approach to WSI (as desired by the WASP philosophy) suggests that standard processes should be used and that if necessary, only small, simple cooling fans (eg. 4 m/s) should be employed. Consequently, it appears that WASP power density is strongly limited by thermal dissipation (with small wafers or natural convection cooling) or power supply (with larger wafers and forced air cooling. In either case, maximum power densities that can be supported seem to be restricted to 0.3-0.4 W/cm^2 at the very most and this power density limit decreases rapidly with larger wafer sizes.

The following section evaluates the power density of the different CAM designs with respect to these limits, to evaluate their suitability for WSI implementation.

8.4 WASP power density

It can be argued that WASP power density is not purely the WASP CAM array power density. As the APP is composed of other blocks (eg. BCL, WCL, MOGL) and that the WASP device comprises other module types (eg. WI, CC, RAM). Furthermore, WASP power may also be affected by the defect patterns (eg. the number of CC and RAM blocks used).

Against this, in Chapter 5, it has been shown that the CAM array in the SCAPE chip is responsible for just over half of the power consumption of the SCAPE processing logic (51%). Optimisation of BCL, WCL and MOGL circuit designs for low power may offer scope for power reduction and hence reduce their contribution to WASP APP power. Furthermore, a proportion of the APP area will be occupied by (low power consuming) data and control busses. Consequently, an estimation of CAM array power density does give an initial idea of the sort of power densities that can be expected in WASP. Although it must be

stressed that this analysis is only an initial attempt. When more data on the BCL, WCL and MOGL designs of the WASP APP is available, the analysis can be refined.

The following sections calculate the WASP power density based on CAM power and look at the implications this has on power rail area and thermal management.

8.4.1 Assumptions

This section details the assumptions made in the estimation of WASP power density using different associative memory designs.

(1) 8192 APEs are active, other APEs are powered down (ie. unused pseudo-static CAM cells are not refreshed).

(2) As in the SCAPE chip, each APE comprises a 37-bit CAM word.

(3) As proposed in the WASP IP device, the 8192 APEs are distributed between 196, $5mm^2$ APP modules (49 cm^2 total APP area). Each module comprises a 64-APE APP.

(4) CAM power is as detailed in Table 6-10, Chapter 6.

(5) Wafer power density is approximated to

$$\frac{\text{CAM power x CAMs/wafer}}{\text{APP area}}$$

8.4.2 Experimental results

Using the above assumptions the total wafer power and wafer power density is summarised in Table 8-2.

Figure 8-6., compares the power density of a WASP device using 5 different CAM designs against the power density limits of natural convection cooling and 20% power rail area. It can be seen that CAMs A and B exceed the natural convection limits by a significant margin for all wafer sizes. However, designs C, D and E all meet the power dissipation limit.

Name	Wafer Power (W)	Wafer power density (W/cm^2)
CAM A	19.6	0.40
CAM B	16.3	0.33
CAM C	3.2	0.06
CAM D	3.4	0.07
CAM E	3.4	0.07

Table 8-2. 4-inch Wafer power and wafer power density

8192 x 37 CAM cells and 196, $5mm^2$ APP blocks.

Figure 8-7., similarly compares the power density with 4m/s air cooling and 20% power rail area. Here, the situation is altered. Smaller (eg. 3 inch) wafers could use any design, as both the power dissipation and the rail area appears acceptably low. However, implementation in 4 inch wafers, results in CAM A still meeting power dissipation limits, but the power rail area is getting slightly over 20%. With wafer sizes over 5 inches even if the power rail area could be supported, the thermal dissipation limit is exceeded.

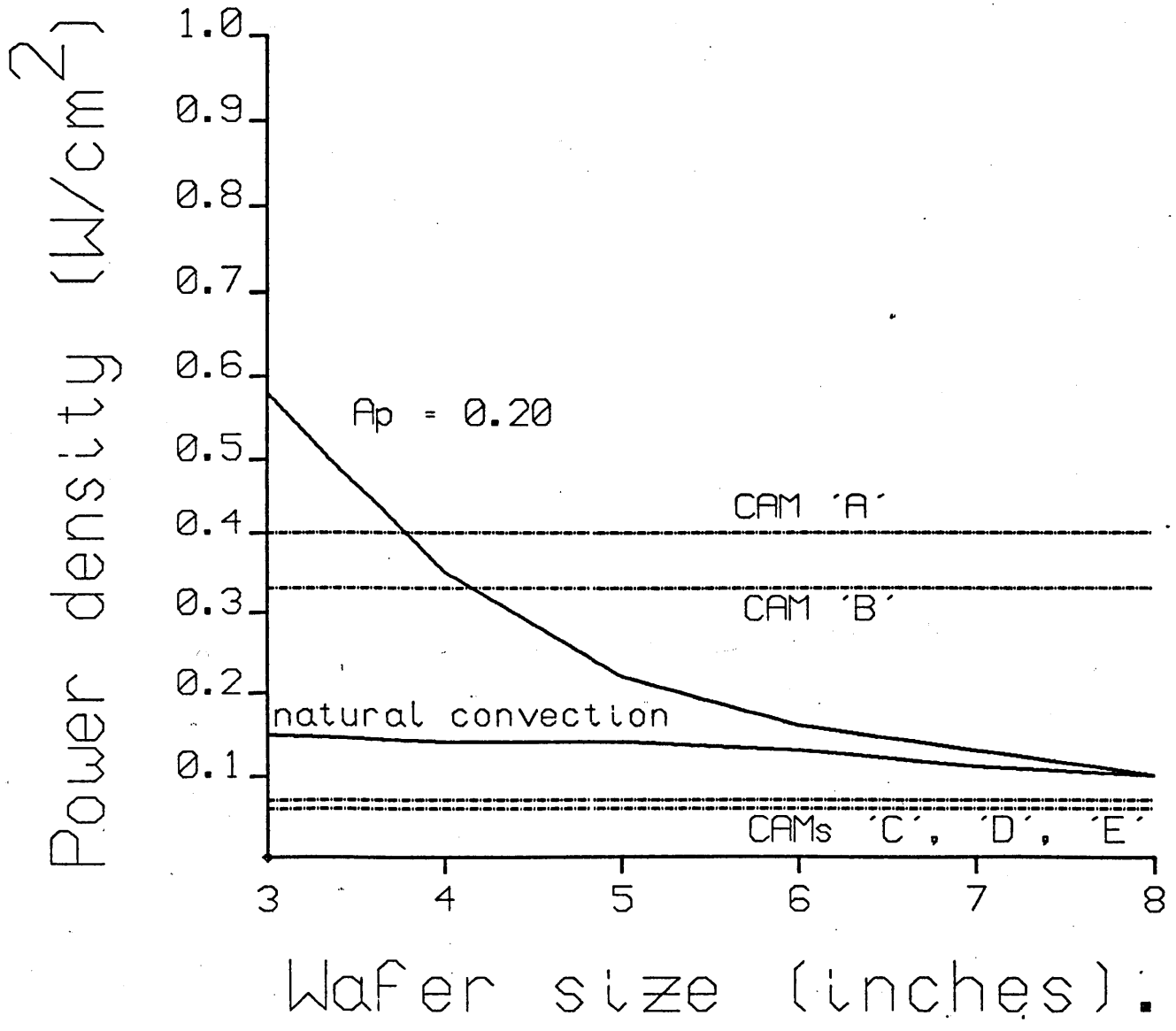


Figure 8-6. WASP power density for CAMs A-E, natural convection, 20% power distribution area.

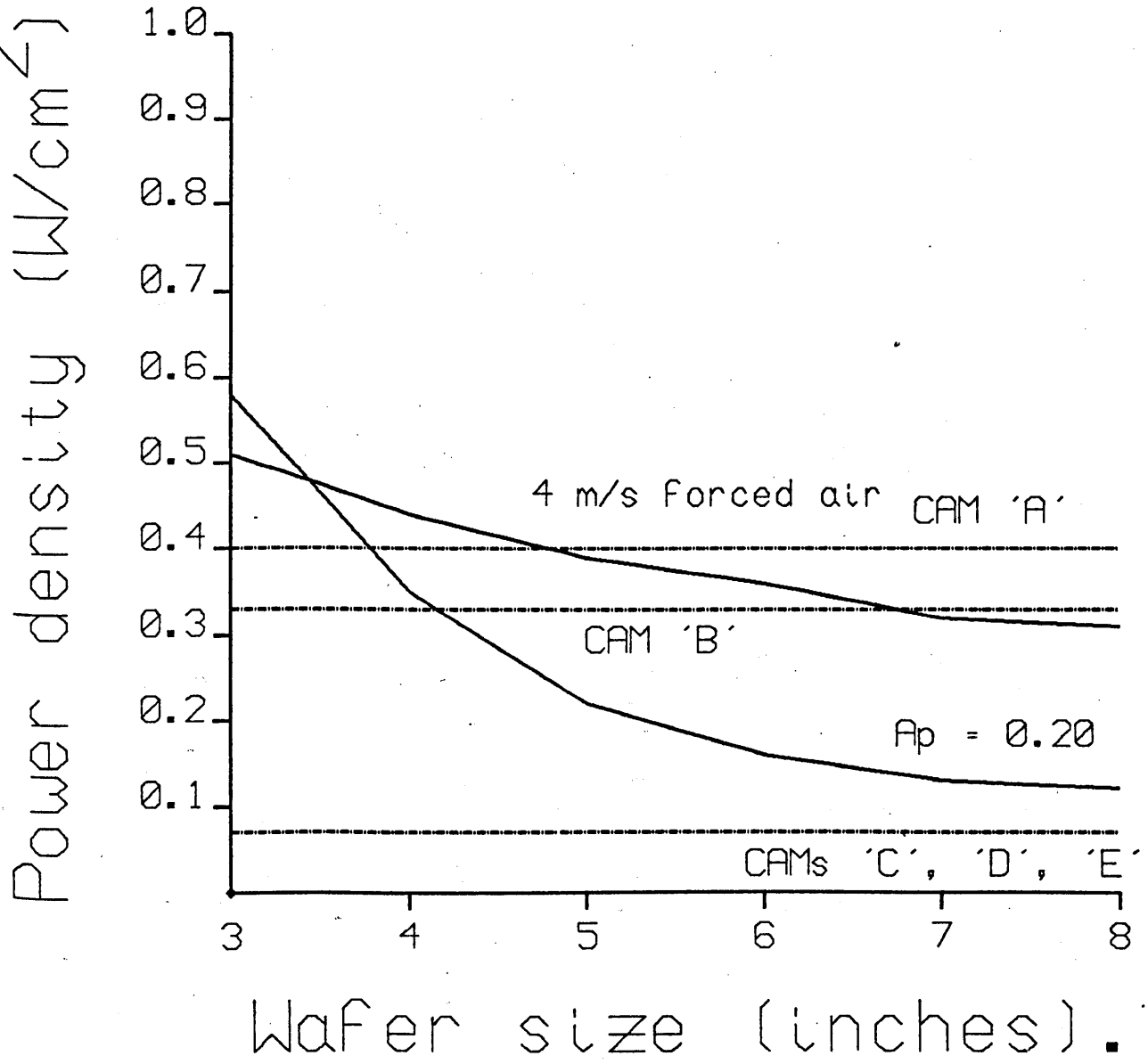


Figure 8-7. WASP power density for CAMs A-E,
4 m/s forced air, 20% power distribution area.

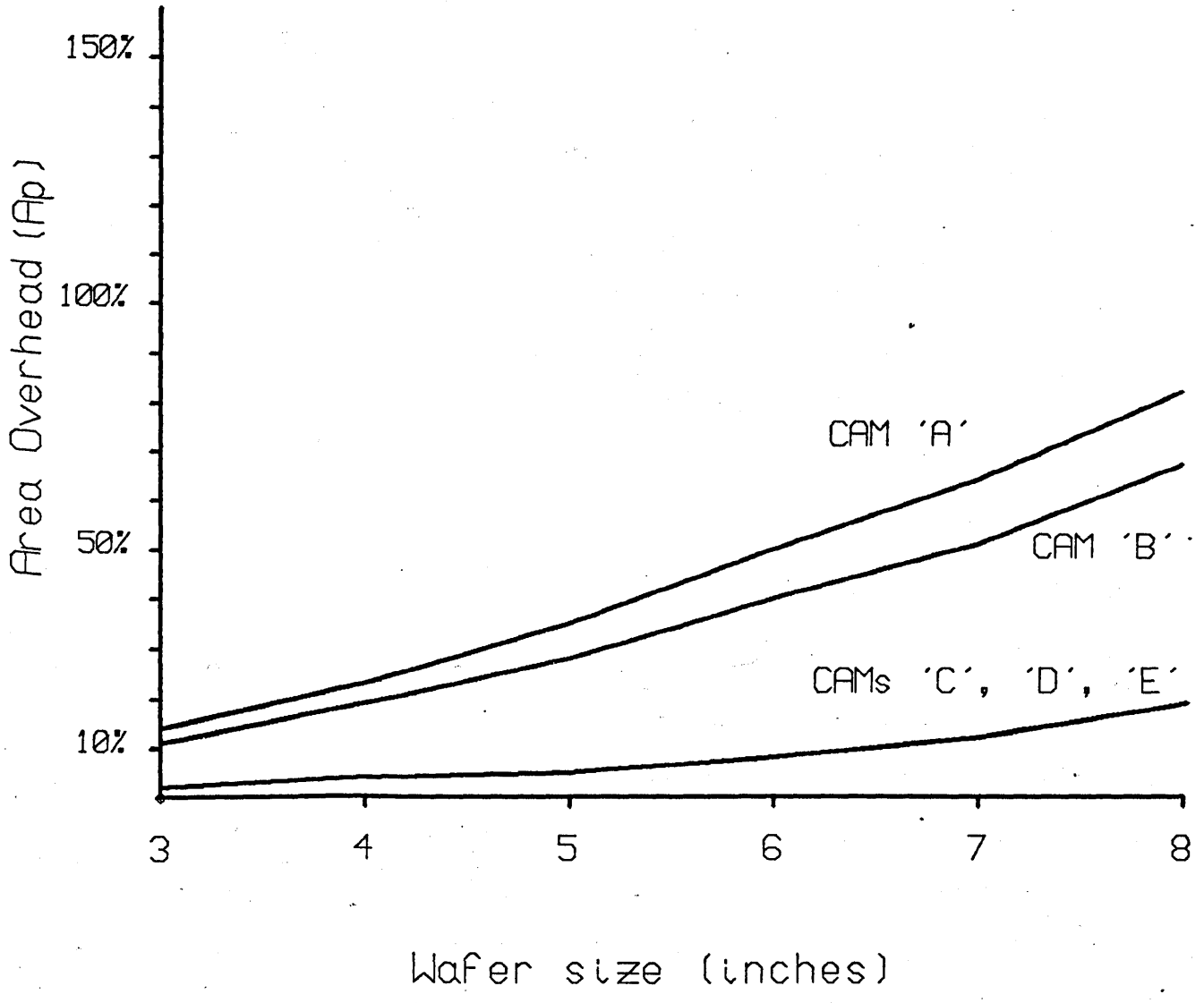


Figure 8-8. Power rail overheads for CAMs A-E.

CAM B by contrast, meets the thermal dissipation limit for all wafer sizes plotted. However, in terms of rail area, the 20% point is exceeded for wafers greater than 4 inches in size.

CAMs C, D and E meet the thermal and power rail criteria for all wafer sizes.

Figure 8-8., compares the power rail overhead (Ap) for a WASP wafer using the 5 different CAM designs. It is interesting to note that CAMs A and B have moderate power rail area requirements with the smaller wafer sizes, but with 6 inch wafers and larger, over half the wafer area is covered with power tracks. CAMs C-E have modest power rail requirements for all the wafer sizes plotted.

8.5 Evaluation - experiment 4

It is clear that the dual constraints of thermal dissipation and power supply severely restrict WSI power densities to about an order of magnitude lower than comparable VLSI chips. Of course, there are a number of technological factors that can be altered, such as the proportion of area occupied by metal (or equally, the thickness of the metal tracks or the number of metal layers) which if increased significantly would restore thermal dissipation as the limiting factor. In addition, more sophisticated cooling mechanisms (eg. liquid cooling) could be employed (and the limit then becomes power supply once more).

To significantly raise WSI power densities therefore requires both a low-resistance metallisation layer (or, extra metal layers) and a water or freon type cooling mechansism. In view of the costliness of these approaches combined with WSI not yet having demonstrated its

commercial potential, it appears that contemporary WSI architectures must use currently available VLSI processes and simple cooling mechanisms and therefore must have very low (viz. an order of magnitude lower than VLSI) power densities to be feasible.

Looking at the Figures 8-6. to 8-8., it can be seen that the calculated wafer power densities of CAMs A and B suggest they can be used with the smaller wafer sizes (viz. 3 and 4 inch slices). However, at the larger wafer sizes, their power densities are such as to make it difficult to remove power without using expensive cooling systems and devoting a significant proportion of metal tracking to power supply.

The more sophisticated CAMs C-E have very low power densities and hence need only natural convection cooling and low power overheads for all wafer sizes.

Thus for the smaller wafer sizes, the more compact CAMs A and B might appear most attractive (although a cooling fan will be necessary). But for the larger wafer sizes, or for all wafers that use natural convection for cooling, it seems essential that the more complex CAMs C, D and E need to be used. The penalty for using these CAMs is increased silicon area and hence a potential reduction in the number of APEs that can be integrated.

The following experiment aims to identify the PE packing density that can be achieved with the different CAM designs.

8.6 Investigation Strategy - experiment 5

The current status of the proposed WASP device makes it inappropriate

to forecast the precise number of PEs that can be integrated. Rather, the investigation strategy concentrates on the influence that the different CAM designs has on the area of the WASP APP.

8.6.1 Assumptions

The following assumptions have been made in forecasting the area of a WASP APP

(1) In the absence of any other information, the areas of the BCL, WCL and MOGL are assumed to be the same as in the SCAPE chip (Table 8-3.).

(2) Each WASP APP comprises 64 APEs.

(3) Each APE comprises a WCL slice and 37 CAM cells.

(4) WCL, BCL, and MOGL area is assumed to be invariant for all 5 CAM designs.

(5) The WASP APP module does not need pads.

8.7 Experimental results - experiment 5

Table 8-4. gives the AMA, APE and APP area for a 64-APE WASP APP, using the 5 different CAM designs (namely, CAM A - Figure 6-2., CAM B - Figure 6-3., CAM C - Figure 6-4., CAM D - Figure 6-5. and CAM E - Figure 6-6.). Table 8-5. gives the same data relative to a base of CAM A.

From these tables, it can be seen that the AMA sizes, as expected, grows in direct proportion with the CAM cell sizes (cf. Chapter 5, Table 5-7.). However, as an APE comprises an AMA word row and a slice

of WCL, the impact it has on the APE size is somewhat less. As an APP comprises a string of APEs together with bit-control logic and a MOGL, The effect of the CAM design on overall APP size is even smaller.

Module	Length (uM)	Height (uM)	Area (uM^2)
BCL	46	2560	117,760
WCL	36	2000	72,000
MOGL	4540	2560	11,622,400

Table 8-3. APP layout data (from [8]).

Name	Area (mm^2)		
	AMA	APE	APP
CAM A	4.60	9.21	25.19
CAM B	7.05	11.66	27.64
CAM C	9.85	14.46	30.43
CAM D	13.02	17.63	33.61
CAM E	12.41	17.02	33.00

Table 8-4. WASP: AMA, APE and APP areas.

8.8 Evaluation - experiment 5

The evaluation of these results is divided into 2 sections: firstly, the impact on general WASP architectures and secondly, the specific impact on the proposed 4-inch WASP IP device.

Name	Relative area		
	AMA	APE	APP
CAM A	1.00	1.00	1.00
CAM B	1.53	1.27	1.10
CAM C	2.14	1.57	1.21
CAM D	2.82	1.91	1.33
CAM E	2.69	1.84	1.31

Table 8-5. WASP: relative AMA, APE and APP areas.

8.8.1 General evaluation

Tables 8-4. and 8-5. have shown that while the more complex CAMs have 2-3 times the area of the simplest CAM (CAM A), they have a significantly smaller impact on overall APP size. This is encouraging, as it means that while the CAMs consume most of the APP power, (51% in the SCAPE chip) they occupy proportionately less of the total chip area. Consequently, increasing the complexity (and hence AMA area) of the CAM to reduce power consumption, does not increase APP area in the same proportion. Of course, if the areas of the BCL, WCL and MOGL change, either due to a change in specification, or as with the CAM design, to reduce power consumption, then this may no longer be the case.

Having said that, it is clear from Table 8-4. that CAM A gives the smallest WASP APP size. However, the power density of an APP using CAM A is such as to probably make it only suitable for WSI with small wafer sizes combined with fan assisted cooling.

The static CAM B results in a 10% increase in overall APP area, but still is only suitable for 3-4 inch wafers and fan assisted cooling. On this basis then CAM B. would seem less attractive to WSI implementation than CAM A. A perhaps suprising result.

The CAM that gives the smallest APP size which also meets the thermal and power limits for all wafer sizes is CAM C (APP 21% larger than an APP based on CAM A). CAMs D and E result in APPs which are some 31-33% larger than CAM A.

8.8.2 WASP IP evaluation

The proposed WASP IP currently specifies a $25mm^2$ reticle incorporating 64-APEs on a 4 inch wafer.

CAMs A and B are the only designs that give an APP size close to $25mm^2$. CAM A is only 0.19 mm^2 oversized. However, the power density of these designs is such that they cannot be used with natural convection. Indeed, a 4 m/s fan is necessary to remove the heat generated. However, with a 4 inch wafer, the power track area overhead is over 20% with CAM A and just under 20% with CAM B. Consequently, while these designs are closest to the reticle limit, they do require a fan to be used with the WASP IP device and also, for power track area to be larger than in the VLSI ASP SCAPE.

CAMs C, D, and E can all be used with natural convection cooling and low power track areas and hence seem attractive for WASP IP implementation. However, they all exceed the proposed reticle size, by as much as 33% in the case of CAM D. Consequently, assuming the $25mm^2$ reticle area, 64 APEs per APP (and hence 12544 APEs in total

and a 65.3% APE harvest) may be difficult to achieve using designs C, D, and E. However, this does not mean the target is infeasible as alternative design options are available to realise a 8192-APE WASP IP wafer.

(1) Integrate fewer APEs per APP and rely on a higher harvest to achieve an 8192-APE string.

(2) Increase the reticle size (and hence reduce the number of 'chip' sites on the wafer) to fit more APEs per APP thus amortizing the BCL and MOGL logic among more APEs.

(3) Use a larger wafer size, than the 4 inch slice currently targetted, to increase reticle area.

Option (1) requires higher fault-tolerance from the WI, CC and RAM blocks. Option (2) offers room for improvement, but sharing a MOGL and BCL among more APEs may increase the number of APEs lost if the MOGL or BCL fails. Option (3) offers the more straightforward solution, but at the cost of potentially more expensive processing. Although, 5 and 6 inch slices are increasingly becoming the industry standard.

To summarise: APPs using CAMs A and B best meet the proposed 4 inch wafer WASP IP reticle size, but require fan assisted cooling. APPs based on CAMs C, D and E, which have been shown to meet the thermal and power limits, for all wafer sizes, exceed the desired reticle size. This implies that harvest or floorplan modifications may be required to achieve an 8192-APE image processor on a 4-inch wafer with these designs.

8.9 Conclusions

This chapter has been concerned with Experiments 4 and 5 in Chapter 4. Experiment 4, was aimed at determining the suitability of the range of associative memories for implementation in WSI.

Thermal management and power density were identified as being key factors. Thermal management constraints severely restrict the wafer power density that can be supported using natural convection. Air assisted fans can improve this but the maximum power density is well below VLSI limits.

However, using fan assisted cooling, only improves the maximum power density for the smaller wafer sizes (eg. 3-4 inches). For once this limit is raised, the dual problem of feeding power into the wafer arises with the larger wafer sizes (eg. 5-8 inches). The problem of excessive area occupied by power tracking is the more difficult to solve technologically, due to the cost of process modifications for low-volume products.

A formula has been derived to relate wafer power density to the area required for the power distribution tracks. Applying this formula for a range of wafer sizes, demonstrated that to implement WSI systems using current VLSI processing technology, power densities may need to be an order of magnitude lower than in VLSI in order that power track area is acceptably low.

CAMs A and B have a power density in excess of the natural convection cooling. If fan assisted cooling is used, CAMs A and B require very large power track areas for wafer sizes larger than 5 inches.

CAMs C, D and E meet both the power track and thermal management

limits for all wafer sizes.

Experiment 5, investigated the impact the different CAM designs has on APP area. It was shown that a large increase in CAM area, only moderately increased APP area. The power density of CAMs A and B is such as to probably make the better suited to 3, or possibly 4 inch wafer sizes. The CAMs C, D and E while easily meeting the thermal and power rail constraints, produced noticeably larger reticle sizes.

In terms of the WASP IP device, CAM A gave an APP size close to the required reticle size. However, CAM A requires a 4 m/s fan assisted air flow for thermal management and slightly over 20% power track area. CAM B gave a 10% larger reticle size at the cost of a slightly smaller power track area overhead. CAMs C (which gave the smallest APP - 21% larger than CAM A), D, and E, imply natural convection and a low power track overhead. However, the increased APP size may require some adjustment in WASP IP floorplanning.

Overall, these results imply that wafer power density is a major design problem in WSI. However, circuit designs exist that can satisfy these constraints. Use of these circuit designs implies a noticeable increase in overall reticle area requirements.

The following and final chapter arrives at the main conclusions from the whole of this thesis, critically assesses the research project, suggests areas for future research and identifies overall, how well the thesis meets its objectives.

CHAPTER 9 CONCLUSIONS

9.1 Objectives

This chapter has the following objectives

(1) Draw together the experimental results and arrive at the main conclusions of the thesis.

(2) Critically assess the research project.

(3) Suggest areas for future work.

9.2 Discussion of results

This section discusses the results of this thesis and the implication that these results have on the potential of WSI for implementing low-cost fine-grain parallel processing computer systems.

9.2.1 Review of objectives

The objective of this thesis is to investigate the potential of WSI for the implementation of low-cost fine-grain parallel processing computer systems.

This can be considered as comprising 2 sub-objectives, namely

(1) Explore the design constraints of WSI and hence investigate the implications these constraints have on the engineering of fine-grain parallel processor systems in WSI.

(2) Identify suitable design techniques for implementing fine-grain WSI parallel processor computer systems.

9.2.2 Review of strategy

The research strategy taken was to investigate the potential of WSI through an investigation into the implementation of a representative architectural component using a promising WSI architecture as a research vehicle.

The thesis addressed WSI implementation issues through an in-depth investigation into the feasibility of implementing a suitable associative memory for WASP, that met the demanding technological constraints of WSI.

The investigation strategy was broken down into 5 seperate experiments, namely

(1) Experiment 1 - associative memory characteristics. This involved the investigation of the electrical, physical and control characteristics of a range of associative memory designs.

(2) Experiment 2 - evaluation in ASP environment. This involved investigating how these designs performed within the operational environment of an image processing ASP.

(3) Experiment 3 - associative memory yield implications are evaluated by investigating the harvest and yield implications of using the associative memory designs in the WASP device.

(4) Experiment 4 - WSI implementation issues. This involved evaluating how well the WASP device, using different CAM designs, would meet the electrical and physical design

constraints of WSI.

(5) Experiment 5 - PE packing density. The influence the different CAM designs has on WASP PE packing density is analysed in this experiment.

9.2.3 Achievements

The thesis has explored some of the design constraints of WSI and the implications these constraints have on the engineering of fine-grain parallel processor computer systems in WSI.

The critical appraisal of Chapter 2 has analysed the past work on WSI architectures. It has pointed out the importance of using contemporary VLSI technology for low-cost implementation. The problem of implementing a silicon-efficient fault-tolerance strategy has been noted. Hierarchical fault-tolerance is argued to be a cost-effective approach. This has been borne out in the WASP multi-level tree fault-tolerant strategy (Chapter 4) which has a good harvest for a modest silicon overhead.

Chapter 7 investigated the yield implications of using the different CAM designs in WASP. Perhaps surprisingly, it showed that the fine-granularity 'n out of m' fault-tolerance of the APEs made the structure relatively insensitive to CAM design. For example CAM D which increased APE area 1.91 times over the smallest APE option (using CAM A), resulted in a reduction in the total number of functional APEs of only 5% compared with an APE using CAM A.

Chapter 2 also highlighted that there has been little work done on the electrical and physical implementation aspects of WSI fine-grain parallel-processors. Chapter 4, having identified from a wider

PAGE MISSING IN ORIGINAL

operations.

However, to achieve suitably low power densities, WSI designs will not be able to afford this tradeoff. An example of this is CAM 'C', which uses extra transistors, to isolate the data part of the CAM cell during write 'X' operations.

A consequence of this increased circuit complexity, is that 'WSI' circuits tend to be larger than 'VLSI' circuits. This means, that for a given silicon area, less circuits can be integrated in WSI, than in VLSI. In view of the fact that even the smallest 'WSI' CAM (CAM 'C') is some 2.14 times larger than the smallest 'VLSI' CAM (eg. CAM 'A', a variant of which was selected for implementation the SCAPE chip), this could be a major problem. However, this problem is eased by the fact that the power density of different circuits varies significantly (eg. in Chapter 6, the SCAPE CAM consumes some 51% of the total processing power, but only occupies some 22% of the processing logic area). Consequently, as Experiment 5 shows (Chapter 8), by optimising for power density on the most power intensive circuits, it is possible to reduce the total area increase (eg. CAM 'C', which is 2.14 times larger than CAM 'A', results in an APP which is only 1.21 larger than an APP that used CAM 'A'). Nonetheless, a noticeable area overhead is required.

The second objective of this thesis was to identify suitable design techniques for fine-grain parallel processor computer systems.

If the associative memory results are applicable to other circuit structures, then the design techniques must consider the power consumption of their circuits. As some of the CAM designs have too high a power density for cost-effective implementation in larger

wafer sizes.

However, as Experiments 1 and 2 show, it is not possible to identify power requirement from a simple circuit simulation. Rather, it is essential to consider the performance of the circuit within its expected operational environment. For example, in Experiment 1, Chapter 6, CAM 'A', has a much lower write '0/1' power (51 uW) than CAM 'C' (165.4 uW). However, when evaluated within the image processing operational environment (Experiment 2, Chapter 6) CAM 'A' turns out to have a power consumption 6.1 times (64.7 uW) larger than CAM 'C' (10.6 uW).

Therefore, in the design of fine-grain parallel processing WSI systems, it is important to evaluate the design within the operational environment.

Although, a rider should be added to this. All 5 CAM designs, could be used with 3 and 4 inch wafers (although designs 'A' and 'B' need a cooling fan), it is only when larger wafer sizes (eg. 5 inches and larger), do the power densities overheads most clearly demand the low-power designs, such as CAMs 'C', 'D' and 'E'.

Consequently, it may be possible to use compact designs for small wafers, but if wafer sizes that are currently being applied (eg. 6 inch slices) are considered, then one must closely design the circuitry to match the operational environment so as to provide an acceptable power density.

Furthermore, the power track area formula developed in Experiment 4, can be used as part of the design techniques, to forecast the power track area requirement of different architectural and circuit

designs.

9.3 Criticism of thesis

The first criticism of the thesis is that the range of the investigations has been neccessarily limited by the 3 year research project. Only 1 architecture has been considered, and in-depth only 1 aspect (albeit a major component) of the architecture has been investigated.

Furthermore, the thesis has not considered other design issues in WSI. For example, packaging and testing issues have not been fully explored.

In the evaluation of CAM performance, only one operational environment was considered, other environments may alter this data.

The yield implications investigated in Experiment 3, used simplistic yield models.

Experiment 4 made some simple assumptions as to the calculation of WASP power density, these assumptions may not be highly accurate, when the WASP design is complete. Similarly, Experiment 5 made simple assumptions as to the area of the WASP APP BCL, WCL and MOGL. Again, these assumptions may be inaccurate when the WASP APP is fully developed.

On a wider scale, throughout the experiments, a 2-micron bulk-CMOS process has been assumed, alternative processes or technologies may alter the balance of results.

Finally, this thesis has been concerned solely with whole wafer devices, no evaluation of WSI hybrids (eg. using a wafer as a

substrate and down-bonding VLSI/ULSI chips) or part-wafer (eg. ULSI) devices has been performed.

9.4 Future work

This section aims to discuss areas of future work relevant to evaluating the potential of WSI for implementing parallel processsors, and in particular, aspects for further research stimulated by the thesis.

Firstly, there clearly needs to be more work done before the potential of WSI for implementing low-cost fine-grain parallel processor computer systems is fully understood.

In particular, alternative architectures need to be considered to more fully explore the tradeoffs and cost-effective constructs in WSI parallel processor design.

The thesis while having gained in terms of in-depth analysis, may have lost out on a more broad-based approach, that considered the interaction between the WI, CC, RAM and APP modules. Clearly, a more developed analysis of the WASP architecture should be undertaken (Using the thesis as part of the analysis) relating the architectural tradeoffs at a higher level (eg. more RAMs less APPs, or alternative control structures).

At a more detailed level, there is room for improving the number of CAM designs evaluated and investigating, alternative novel circuit styles. Clearly, investigations are also needed into the effect of different operational environments on the CAM cell, APP and WASP design.

As more detailed work on the WASP BCL, MOGL and WCL is undertaken, this can be used to improve the analysis of WASP power and PE packing densities.

Work should also be undertaken into evaluating the effect of alternative fabrication styles (eg. better processes, different technologies) to identify the effect these have on the potential of WSI. Similarly, investigations into the use of WSI hybrids and ULSI designs should be performed, particularly to investigate the cost-effectiveness of implementing the same system in different ways (eg. VLSI, fault-tolerant VLSI, ULSI, WSI hybrid and WSI).

9.5 Conclusions

The thesis has necessarily been restricted in its investigation. Furthermore, not all the experiments have been fully carried out. Some of the experiments have had to use simplistic assumptions. In view of this, what has the thesis contributed to the body of knowledge on the potential of WSI for fine-grain parallel processing computer systems implementation?

The thesis has clearly demonstrated the influence of power density on the successful implementation of WSI parallel processing systems. It has shown that there exists dual constraints of heat removal and power track area. While low-cost options can improve heat removal, improvement of the power track area factor requires process modifications. The thesis has shown that for cost-effective implementation, WSI power densities need to be about an order of magnitude lower than VLSI power densities.

The thesis has also shown that it is feasible to design fine-grain parallel processing circuitry that meets the power density requirement. Furthermore, the overheads involved in using such circuits have been exposed. The thesis demonstrated how to achieve an acceptable power density. This is done by closely matching the design of the circuitry to the operational environment.

These contributions directly relate to the objective of the thesis. However, the thesis has also made some 'indirect' (ie. not directly related to the research objectives) contributions. It has taken a WSI architecture of note and developed it much further. The work on the CAM designs, their operation and control has contributed to the field of associative memory and processor design.

Finally then, WSI does appear to have potential for the implementation of low-cost fine-grain parallel processor systems. Systems such as WASP, may be promising architectures for applications in the areas of image, signal, IKBS and 5^{th} generation high-level language support. However, if this potential is to be cost-effectively realised, the results of this thesis argue for the importance of power density and appropriate circuit design to be considered from the outset.

[1] P.E. CANTER, 'Image processing - a step towards artificial intelligence', Aerospace Dynamics, Issue 14, pp 2-15, British Aerospace Dynamics Group June 1984.

This paper discusses the applications and state of the art of image processing. It outlines the very high computational speeds required for real-time image processing and how a fine-grain array of simple processing elements can support this computational speed. Finally, the relationship between image processing and artificial intelligence is discussed.

[2] M.J.B Duff., 'Review of CLIP image processing system', Proceedings National Computer Conference, pp 1055-1060, 1978.

The paper overviews the structure of the CLIP cell, and the way in which CLIP devices are composed. Examples of simple operations are given. A 96 x 96 (9216 PE) CLIP chip array is described together with expected operational speeds.

[3] K.E. Batcher, 'Design of a massively parallel processor', IEEE Transactions on Computers C-29 ,pp 836-840, 1980.

Describes the 16384 PE MPP processor system based on the 8-PE MPP chip. The MPP is organised as an array but has programmable edge connectivity. Redundancy is built-in through a spare row/column mechanism. There is extensive control and staging memory support of the MPP array.

[4] J.B.G. Roberts, P. Simpson, B.C. Merrifield, J.F. Cross, 'Signal processing applications of a distributed array processor', Proceedings IEE Computers and Digital Techniques, 131 No 6 1984.

The use of the DAP architecture for signal and image processing is described. Also, characteristics of the MILDAP chip are outlined. Processing speeds for the MILDAP chip performing FFTs are reported.

[5] M.M. McCabe, A.P. McCabe, B. Arembepola, I.N. Robinson, A.G. Corry, 'New algorithms and architectures for VLSI', GEC Journal of Science and Technology, Vol 48, No 2, 1982.

This paper discusses the impact VLSI has on the design of computer architecture. The suitability of systolic arrays is shown and the design of a general purpose GRID chip, a bit-serial array processor in VLSI, is discussed.

[6] T. Sudo, T. Nakashima, M. Aoki, T. Kondo, 'An LSI adaptive array processor', Proceeedings IEEE International Solid-state Circuits Conference, pp 122-123 1982.

Briefly describes the NTT AAP (adaptive array processor) chip, an 8 x 8 matrix of 8-way connected bit-serial PEs. Each PE comprises an ALU and some register storage. Power dissipation is 1.6W.

[7] R. Dettmer, 'Chip architectures for parallel processing', IEE Electronics and Power, pp 227-231 March 1985.

Evaluates a range of chip architectures for fine-grain parallel

processing. Concentrates on the GAPP chip and discusses its use in a range of applications. Notes that the high-pinout of the GAPP array architecture makes the chip quite expensive.

[8] I.P. Jalowiecki, R.M. Lea, 'SCAPE - A high performance image and signal processor chip', Proceedings 1986 Silicon Design Conference, Wembley UK. 1986.

Describes in detail the SCAPE chip architecture and application areas. Specifically, focusses on an analysis of the electrical design issuses of the SCAPE chip. Identifies the associative memory as dominating SCAPE power density.

[9] R.M. Lea, 'SCAPE: a single-chip array processing element for signal and image processing', Proceedings IEE Computers and Digital Techniques, 133, pp 145-151 1986.

Lea's paper describes the design and implementation of the SCAPE APP chip. Design considerations and a description of the various functional elements are given. Testability and design verification aspects are discussed.

[10] R.M. Lea, 'VLSI and WSI associative string processors for structured data processing', Proceedings IEE Computers and Digital techniques, 133, pp 153-162 1986.

Details the architecture, software and VLSI implementations of associative string processors. Outlines an algorithm for pattern matching and discusses ASP performance. Describes the WASP project for implementing WSI-ASPs.

[11] R.M. Lea, 'Associative processing', (Advanced Digital Information systems, Ed. I. Aleksander), Prentice Hall, pp 531-575, 1985.

This chapter, provides an analysis of associative processing principles and the application areas where it is of use. An overview of current VLSI and WSI ASP hardware projects is given.

[12] S.R. Jones, S.J. Hedge, A.R. Barney, R.M. Lea, 'Preliminary specification of the SCRIPT chip', Brunel University Technical Memorandum No CM/R/138, April 1985.

This document comprises an outline proposal for the SCRIPT chip, an APP optimised for text processing, IKBS and 5^{th} generation language support. Details of floorplan, functional units and expected chip size and yield are provided.

[13] S.R. Jones, M.P. Fourman, R.M. Lea (Eds.) 'Proceedings of a Special Workshop on WSI', Brunel University, March 1984.

Reports on a workshop with representatives from Brunel, ICL an Plessey contributing. Concentrates on potential WSI applications and design issues.

[14] E.A. Sack, R.C. Lyman, G.Y. Chang, 'Evolution of the concept of a computer on a slice', Proceedings of the IEEE, Vol 52, No 12, pp 1713-1720, December 1964.

This paper is the first recorded reference to WSI. A 50-gate synchroniser and a 118-gate shift register were built using arrays of NAND gates and discretionary wiring. The limit they

faced was in building a suitable package. Eventually it was hoped to implement a complete ALU on a wafer.

[15] R.L.Petritz, 'Current status of large scale integration technology', IEEE Journal of Solid-state Circuits, No 4, pp 130-146, December 1967.

Reviews the then state of the art in LSI technology. Strongly advocates the use of WSI (or full-slice technology as it was then called) for the most cost-effective use of silicon.

[16] J.W. Lathrop, R.S. Clark, J.E. Hull, R.M. Jennings, 'A discretionary wiring system as the interface between design automation and semiconductor array manufacture', Proceedings of the IEEE, Vol 55, No 11, pp 1988-1997, November 1967.

This paper gives an in-depth report on the use of discretionary wiring. Cells are probed and a wafer map created. A discretionary metal mask is designed using a CAD suite to link good cells together.

[17] D.F Calhoun, 'The pad relocation technique for interconnecting LSI arrays of imperfect yield', Proceedings of the AFIPS Fall Joint Computer Conference (FJCC) 1969, Vol 35, pp 99-109, 1969.

Describes pad relocation, a technique for amortising the cost of one discretionary mask among many wafers. A core set of cells are identified and a mask created to link them together. Most wafers have a core set of cells working and hence use this mask. Those that do not, have their pads relocated to nearby good

cells, by a small simple custom mask.

[18] D.F. Calhoun, L.P. MacNamee, 'A means of reducing custom LSI interconnection requirements, Journal of Solid-state Circuits, Vol 7, No 10, pp 395-404 October 1972.

Another pad relocation paper, reporting progress made with the technique. A mathematical treatment of the mask costs are given which suggest the pad relocation technique offers a cost-effective approach to discretionary wiring. Details of a 620-gate 1.5 inch wafer are given.

[19] C.A.Finilla, H.H. Love, 'The associative linear array processor', IEEE Transactions on Computers, Vol C-26, No 2, pp 112-125, February 1977.

ALAP was the first WSI processor to be implemented. A distributed-logic memory (DLM) processor, using discretionary wiring. In addition, reliable fault-isolation circuitry is incorporated to identify any post-discretionary faults. A 2 inch ALAP wafer was fabricated and applied to arithmetic and radar tracking problems.

[20] F.B. Manning, 'An approach to highly-integrated computer-maintained cellular arrays', IEEE Transactions on Computers, Vol C-26, No 6, pp 536-552, June 1977.

Manning's approach to WSI uses conventional IC technology and simple programmable function cells. An arm growing technique maps out good cells. The desired structure is configured and cells programmed to the required function. Strings appear to be

a much better fit than arrays.

[21] R.C. Aubusson, I. Catt, 'Wafer-scale integration - a fault tolerant procedure', IEEE Journal of Solid-state Circuits, Vol SC-13, No 3, pp 339-344 June 1978.

This approach shares some similarities with Manning's approach. An arm of good cells is grown on the wafer. However, unlike Manning's, approach, the string is the final structure used. Furthermore, the cells are fixed function and comprise shift registers for data storage.

[22] Y. Kitano, S. Kohda, H. Kikuchi, S. Sakai, 'A 4-Mbit full-wafer ROM', IEEE Journal of Solid-state Circuits, Vol SC-15, No 4, pp 686-693, August 1980.

A 3 inch wafer ROM was designed at NTT laboratories, organised hierarchically into 4 x 1 Mbit modules, for Kanji recognition applications. Each memory cell is duplicated for reliability and off-wafer reconfiguration logic provides coarser-grain fault-tolerance.

[23] Y. Egawa, T. Wada, Y. Ohmori, N. Tsuda, K. Masuda, 'A 1-Mbit full-wafer MOS RAM', IEEE Journal of Solid-state Circuits, Vol SC-15, No 4, pp 677-691, August 1980.

A 3 inch wafer has been fabricated, that uses hierarchical fault-tolerance. Bit and 32-word block substitution are used via external jumper leads. The wafer dissipates 4.7W.

[24] In 'Wafer-scale integration', Eds C.R. Jesshope, W.R. Moore, pp 137-139, Adam Hilger 1986.

Reviews the WISPER1 RAM solid-state disk based on the Aubusson/Catt methodology [21]. WISPER1 is a 0.5 Mbyte serial memory targetted at the provison of fast bulk storage for personal computers.

[25] N.H. MacDonald and G.B. Neish, 'An algorithmic approach to the testing of a wafer-scale integrated (WSI) circuit, Digest of Papers 1982 International Test Conference, pp 590-600, 1982.

This paper reports the results of using the Aubusson/Catt methodology on a 3 inch wafer. Spirals of up to 200 chips were grown and the project intends to experiment with larger wafer sizes.

[26] B.R Elmer, W.E. Tchon, A.J. Denboer, R. Frommer, S. Kohyama, K. Hirbayashi, I. Nojima, 'Fault-tolerant 92160 bit multiphase CCD memory', Proceedings International Solid-state Circuits conference, pp 116-117, 1977.

While this device is not a full wafer, its size (440 x 570 mil) requires fault-tolerance to be supported. A shift register is used to connect working 2560 blocks of CCD memory to form a string.

[27] H. Barsuhn, 'Functional wafer - a new step in LSI', European Solid-state Circuits Conference, pp 79-80, Sept 1977.

Using a 2.5 inch wafer, an 8 K-words x 9 bit memory is

implemented. Defective memory blocks are repaired by down bonding a flip-chip.

[28] D.C. Shaver, 'Electron-beam techniques for integrated circuit testing and customisation', Proceedings IEEE Custom Integrated Circuits Conference, pp 606-609, 1983.

This paper reports on how an E-beam can be used for circuit testing and customisation. E-beam programming is a reversible technique for configuring structures. Furthermore, it can be used to sense and alter logic states. Its feasibility is shown applied to a 128K-bit EPROM.

[29] D.C. Shaver, R.W. Mountain, D.J. Silversmith, 'Electron-beam programmable 128K-bit wafer-scale EPROM', IEEE Electron Device Letters, Vol ED-14, No 5, pp 153-155, May 1983.

A 128K-bit EPROM, that uses a hierarchical redundancy system on a 2 inch wafer. Each 16 k-bit modules comprises 16 1k-bit subsystems. Each subsystem comprises 34 rows of which 32 need to work. An E-beam is used to switch in the required set of modules.

[30] G.D. Chesley, 'Taking a new-look at wafer-scale integration'. Computer Design, pp 157-158 May 1984.

Chesley proposes implementing a large mainframe system on wafer. The storage operates as a virtual memory, faulty blocks being considered as locked pages. Hard configuration using fusible links is recommended. Redundant CPUs are also proposed for implementation.

[31] D.L. Peltzer, 'Wafer-scale integration: the limits of VLSI', VLSI Design, pp 43-47, September 1983.

Describes the Trilogy approach to WSI. The wafer is to implement an IBM 3081 mainframe on a single wafer. ECL technology and fusible links are used. Each wafer dissipates 1.2KW; pin out is over 2000.

[32] R.R. Johnson, 'The significance of wafer-scale integration in computer design', Proceedings of the ICCD: VLSI in Computers conference, pp 101-120, 1984.

Johnson's approach is to use WSI as a packaging technology. The wafer comprises a manhattan mesh of wires with vacant bond sites. Good VLSI chips are then downbonded. Electrical links are blown (or laser zapping used) to achieve the desired connectivity.

[33] J.I. Raffel, 'The RVLSI approach to wafer-scale integration', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 199-203, Adam Hilger 1986.

Introduces the RVLSI technology, a technique that uses lasers to configure a system of MSI/LSI complexity cells embedded in a manhattan mesh of 2-layer metal interconnect.

[34] G.H. Chapman, 'Laser-linking technology for RVLSI', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 204-215, Adam Hilger 1986.

Details of the technology used to make break metal links in

RVLSI are given here. A low power chemical laser is used. This paper demonstrates the reliability of the RVLSI laser process.

[35] S.L. Garverick, E.A. Pierce, ´ A single wafer 16-point 16-MHz FFT processor´, Custom Integrated Circuits Conference (CICC), pp 244-283, 1983.

Describes an FFT wafer made through the RVLSI technology. 30% of the wafer area is occupied with metal tracking and 52% of the multiply-accumulate cells that form the heart of the FFT wafer are harvested.

[36] F.M. Rhodes, ´Application of RVLSI to signal processing´, in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 223-235, Adam Hilger 1986.

Describes 3 RVLSI wafers: the FFT detailed in [35], a digital integrator and a constant false alarm indicator. These circuits contain about 130,000 transistors and require about 1500-2000 links to be treated with a laser.

[37] K.Rose, E.H. Rogers, T.J. Wilson, ´Testability and WSI design´, Proceedings IEEE Curriculum for Test Technology conference, pp 94-100 1983.

This paper describes the Renneslaer Polytechnic approach to WSI. Heavy reliance on CAD for auto routing and placement is a feature of their approach. The design of an FFT processor and a database machine is discussed here.

[38] W.R. Moore, M.J. Day, 'Yield enhancement of a large systolic array chip', Microelectronics and Reliability, Vol 22, No 3, pp 511-526, 1984.

This paper compares a number of schemes for supporting fault-tolerance on a large area IC. A range of bypass techniques were simulated. The most effective approach was one where a faulty module enabled its own bypass.

[39] W.R. Moore, R. Mahat, 'Fault-tolerant communications for wafer-scale integration of a processor array', Mircoelectronics and Reliability, Vol 25, No 2, pp 291-294, 1984.

This paper looks at three techniques for creating good 2-D arrays form a partially faulty circuit. A scheme whereby a faulty cell results in all rows (or columns) 'leftwards' of the fault, shift one place to the right, provides the most cost-effective solutions. However, if there is to be only modest cell redundancy, all these schemes require high cell yields.

[40] H.T. Kung, M.S. Lam, 'Wafer-scale integration and two-level pipelined implementations of systolic arrays', Journal of Parallel and Distributed Computing, Vol 1, No 1, pp 32-55, 1984.

Discusses an approach to systolic fault-tolerance, the systolic ring. The linear array of cells are connected at the ends. Faulty cells are replaced by clocked delays to maintain synchronisation. Laser configuration of the circuit is proposed.

[41] F.T. Leighton, C.E. Leiserson, 'Wafer-scale integration of systolic arrays', Proceedings IEEE 23rd Annual Symposium on the Foundations of Computer Science, pp 297-311, 1982.

Mathematically analyses techniques for configuring structures on a wafer. The algorithms produced attempt to minimise wire length.

[42] A.L. Rosenberg, 'The Diogenes approach to testable fault-tolerant arrays of processors', IEEE Transactions on Computers, Vol C-32, No 10, pp 902-910 October 1983.

The Diogenes approach uses a bus-structured methodology, with PEs hanging off the bus. By selectively connecting PEs to different bus lines, a range of structures can be created. This paper uses soft switches as a means of linking onto busses.

[43] A.L. Rosenberg, 'Graph-theoretic approaches to fault-tolerant processor arrays', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 10-23, Adam Hilger 1986.

This paper compares three approaches (including Diogenes) to creating WSI processor arrays. A detailed graph-theoretic analysis is given.

[44] R. Schuck and M.Glesner, 'A WSI system for the computation of the two-dimensional fast fourier transform', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 10-23, Adam Hilger 1986.

Discusses a proposed WSI FFT processor. Each PE is embedded in

an array of routing switches. Laser or fuse programming is assumed. Each processsor contains around 100,000 transistors.

[45] J. Fried, ´Wafer-scale integration of pipelined processors´, Proceedings IEEE ICCD: VLSI in Computers, pp 611-615, 1984.

Describes a design methodology for building a variety of pipelined processors in WSI. The general purpose PEs test themselves and link into the network. Provision is made for E-beam confguration of interconnect and isolating the occasonal faulty PE that includes itself into the network.

[46] L. Snyder, ´Overview of the CHiP computer´, VLSI ´81, pp 237-246, 1981

ChiP architectures comprise a homogeneous network of PE´s embedded in a switch lattice. The lattice can be configured to create the desired structure. Faulty elements can be routed around. A range of structures are shown embedded in the CHiP lattice.

[47] K.S. Hedlund, L.S. Snyder, ´Wafer scale integration of CHiP processors´, Proceedings International Conference on Parallel Processing, pp 262-264, 1982.

Discusses the WSI implementation of the CHiP architecture. A hierarchical approach is taken to fault tolerance. Each node comprises a 4 x 3 array of CHiP cells, and a 2 x 2 array configured from these. The occasional faulty node is dealt with by bypassing the column.

[48] K.S. Hedlund, L.S. Snyder, 'Systolic architectures - a wafer scale approach, Proceedings IEEE ICCD: VLSI in Computers, pp 604-610, 1984.

Describes a general strategy for implementing a fault-tolerant array, using a 2-level hierarchy. Each node has to form a 2 x 2 array from a 4 x 3 array. Routing switches link around the occasional faulty node.

[49] K.S. Hedlund, 'The design of a prototype WASP machine', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Describes a prototype VLSI implementation of the WASP (Wafer-scale Systolic Processor) device. The chip comprises 9 PEs in a 40 switch lattice. Functional chips were reported.

[50] A.J. Rushton, C.R. Jesshope, 'The reconfigurable processor-array - an architecture in need of WSI', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 148-158, Adam Hilger 1986.

This paper describes the RPA architecture, a bit-serial processor that can be combined to give bit-parallel processing as the problem demands (eg. floating point addition). A 32 x 32 RPA array is proposed for WSI implementation.

[51] C. Jesshope, L. Bentley, 'Techniques for a wafer-scale RPA', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Discusses the fault tolerance scheme used in the RPA. A two-level hierarchy is used. At the lowest level a column of N PEs is configured from (2N +2) devices. At the higher level, programmable links are used to connect sub-arrays together. Split bond pads linked by a discretionary solder ball are suggested as an economical way of acheiving this.

[52] P. Genestier, C. Jay, G. Saucier, 'A reconfigurable microprocessor for wafer-scale integration', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

HYETI, is a high-reliability microprocessor for WSI implementation. HYETI comprises a set of bit-slices, which may be E-beam programmed to create the working device. Thus, it is feasible to implement an array of high-yielding devices with only modest overheads. The processors lie in a switch array, which may also be E-beam programmed to create the desired connectivity.

[53] G. Saucier, 'HYETI - a high yield defect-tolerant processor', Silicon Design, Vol 3, No 3, pp 12-14, 1986.

Describes the Systolimag machine, based on the HYETI processor targetted at the support of image processing applications.

[54] R.A. Evans, 'Wafer scale integration of systolic and other two-dimensional processor arrays', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Describes the WINNER technique for configuring 2-D arrays. WINNER is a self-organizing system, that communicates via simple 'test' and 'available' lines that implement a protocol to create a 2-D array.

[55] R.A. Evans, 'Self organisation is a WINNER', Silicon Design, Vol 3, No 3, pp 16-17, 1986.

Overviews the WINNER methodology, and gives simulation details as to how WINNER performs with varying cell yields.

[56] R.D. Etchells, J. Grinberg, G.R. Nudd, 'Development of a 3-D circuit integration technology and computer architecture', Proceedings of the Society of Photo-optical Instrument Engineers (SPIE), Vol 282, pp 64-72, 1981.

Describes a most adventurous approach of implementing 3-D WSI circuits. Bit-serial ALUs in WSI, connect through the vertical planes via diffused Aluminium columns and connecting microsprings.

[57] R.C. Aubusson, R.J. Gledhill, 'Wafer scale integration - some approaches to the interconnection problem', Microelectronics Vol 9, No 1, pp 5-9, 1978.

Enhances the analysis of the harvest of Aubusson/Catt's basic algorithm by considering richer connectivity and connecting the edges of the wafer together (it is not specified how this is achieved). Richer degrees of connectivity improve harvest, particularly at lower yields. However, all approaches perform better with cell yields over 50%.

[58] R.M. Lea, M. Sreetharan, 'WSI distributed logic memories', Proceedings of the CALTECH conference on VLSI, pp 187-197, 1979.

Outlines a proposal for WSI-DLMs, instruction and data formats are given, together with operation details of a proposed system.

[59] I. Catt, 'Wafer-scale integration', Wireless World, pp 57-59, July 1981.

Describes the Aubusson/Catt approach to WSI, but extends this principle to introduce the slow and fast line for data transmission. Looping and barelling (viz. cycling of data between a few DLMs for repetitive processing) are suggested as useful mechanisms for processing data.

[60] M. Sreetharan, R.M. Lea, 'Text compression with Property 1a', Brunel University Technical Memorandum, No C/R/087, 1979.

Discusses the application of Property 1a to text compression and shows that it is feasible to support real-time (viz. disk rate) text compression/decompression.

[61] M.J. Shute, P.E. Osmon, 'COBWEB - a reduction architecture', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 199-203, Adam Hilger 1986.

Describes the COBWEB-1 system based on Aubusson and Catt's methodology, but enhances communication by opening the radial links. COBWEB is designed for the support of declarative languages.

[62] P. Anderson, P.E. Osmon, 'Communications and processors in COBWEB', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Discusses the 'packet switched network' used to communicate between processors in the COBWEB system. Packets are routed radially until their address is less than that of the next radial link, whereupon they are routed serially until the correct processor is found.

[63] P. Kelly, M.Shute, 'Cartesian routing and fault-tolerance in a wafer scale multi computer' Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Discusses the communication strategy used in COBWEB-2, and analyses the cell harvest as a function of cell yield.

[64] S.R. Jones, R.M. Lea, 'Interconnection strategies for the WASP device', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 82-91, Adam Hilger 1986.

Discusses the suitability of the tree interconnection strategy for the WASP device. Proposes metrics for evaluating the efficiency of different interconnection strategies. Compares the WASP strategy with other WSI architectures.

[65] R.M. Lea, 'WASP - WSI associative string processor', Silicon Design, Volume 3, No 1, pp 14-15, January 1986.

Describes an application-specific WASP device, an 8192-pixel patch processing module. WASP uses conventional fabrication techniques and is designed for low-cost applications.

[66] R.M. Lea, 'A WSI image processing module', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Describes in detail the architecture and operation of the WASP IP. In addition, outlines the likely advantages of using WSI as an implementation technology for ASPs.

[67] C.Y. Lee, 'Intercommunicating cells: a basis for a distributed logic computer', Proceedings AFIPS (FJCC), 22, pp 130-136 1962.

The original paper on DLM architectures. Describes a DLM cell down to the gate level together with DLM programming. The application area is text retrieval.

[68] B.A. Savitt, H.H. Love, R.E. Troop, 'ASP: a new concept in language and machine organisation', Proc AFIPS (SJCC), 30 pp 87-102, 1967

Discusses the association storing processor (ASP), the ASP is built around the ASP language and designed for its explicit support. The ASP advances the Lee cell by the use of context-addressing, which selects all cells specified in a particular data field (viz. explicit field masking).

[69] J.N. Sturman, 'An iteratively structured general purpose digital computer', IEEE Proceedings Transactions on Electronic

Computers, Vol EC-17, pp 2-9, 1968.

Joel Sturman's approach is to increase the sophistication of the Lee cell by allowing it to store its own instructions and thus eliminating the host computer of previous approaches and hence achieving a greater degree of distribution more akin to an MIMD (or certainly an MSIMD) machine

[70] P.A. Beaven, D.W. Lewin, 'An associative parallel processing system for non-numerical computation', The Computer Journal, Vol 15-4, pp 343-349, 1972.

Details the rewuirement for a symbol processing machine. The cell design is based on Lee's, but its use is focussed on symbol manipulation rather than retrieval.

[71] R.M. Lea, J.S. Wright, 'A novel memory concept for information processing', Datafair Research Papers, Vol 2, pp 413-417, 1973.

Gives the requirement for an efficient symbol manipulation machine. Suggests a memory based on Lee's cell as a basis, but adds modular partioning.

[72] A. Mukhophadyay, 'Hardware algorithms for non-numeric computation', Proceedings IEEE Transactions on Computers, Vol C-28, pp 384-394 1979

Mukhophadyay's work is targetted towards string processing, in particular he focusses on SNOBOL and data retrieval applications. He proposes data streaming off a disk flowing through his array of DLM cells and being rewritten to disk. His

cell design uses an i line to support the SNOBOL anchoring function (used to retrace from mismatching strings).

[73] R.M. Lea, 'Micro-APP a building-block for low-cost high-speed associative parallel processors', The Radio and Electronic Engineer, pp 91-99, 1977.

This paper demonstrates the cost-effectiveness of the micro-APP design. Two implementations are discussed and Lea shows that compared with a pure content-addressable memory chip, the micro-APP offers reduced costs through low pin out and reduced sense and drive amplifiers.

[74] K.D. Warren, M.B.E. Abdelrazik, R.D. McKirdy, R.M. Lea, 'A power distribution strategy for WSI', in Wafer-scale Integration (C.R. Jesshope and W.R. Moore Eds.), pp 54-61, Adam Hilger 1986.

Compares the performance of a range of power distribution strategies for WSI. Notes that an independent rails network offers an attractive combination of fault-tolerance and routability.

[75] K.D. Warren, M.B.E. Abdelrazik, H.S. Bolouri, R.M. Lea, 'Power, clock and signal Distribution in the WASP device', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Describes the power, clock and signal distribution in the image processing WASP device. Demonstrates how the tree interconnection strategy offers many benefits for power, clock

and signal distribution.

[76] R.D. McKirdy, M.B.E. Abdelrazik, H.S. Bolouri, S.J. Hedge, S.R. Jones, K.D. Warren, R.M. Lea, 'Comparative Appraisal of the Implementation of an Image Processing Module using WSI and VLSI Associative String Processors'., Proceedings of an International Workshop on Systolic Arrays, paper L5, Oxford University, July 1986.

Compares the implementation of an 8192-pixel patch processing module using VLSI and WSI ASPs. Notes that the WSI implementation offers a wide range of speed and cost advantages.

[77] S.R. Jones, I.P. Jalowiecki, S.J. Hedge and R.M. Lea, 'A 9-Kbit Associative Memory for High-speed Parallel Processing Applications', accepted for publication (subject to modifications) in the IEEE Journal of Solid-state Circuits.

Reports on the design considerations and implementation of the associative memory in the SCAPE chip. Gives a detailed account of the electrical characteristics of the chosen CAM design.

[78] S.J. Hedge and R.M. Lea, 'Intra-module Fault-tolerant Strategies for the WASP Device', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Evaluates 'n out of m' APE fault-tolerant strategies for the WASP device. Demonstrates that sub-string bypassing offers the most cost-effective silicon and yield tradeoffs. Suggests 4-APE sub-strings as being optimal for the image processing WASP device.

[79] C.H. Stapper, F.M. Armstrong and K. Saji, 'Integrated Circuit Yield Statistics', Proceedings of the IEEE, Vol 71, No 4, pp 453-470 April 1983.

A comprehensive review of IC yield statistics and models. Combining both theoretical and practical yield studies to evaluate yield prediction models.

[80] T.E. Mangir, 'Sources of Failures and Yield Improvement for VLSI and Restructurable Interconnects for RVLSI and WSI: Part 1 - Sources of failures and Yield improvement for VLSI'. Proceedings of the IEEE, Vol 72, No 6, pp 690-708, June 1984.

A comprehensive evaluation of yield and failure mechanisms for ULSI and WSI circuits. Reports on the effect of interconnection area on yield improvement, Argues that interconnect failure can often be treated as module failure (eg. by switching in another module), rather than as 'hardcore' (unrepairable) failure.

[81] W.R. Moore, 'Review of Fault-tolerant Techniques for the Enhancement of Integrated Circuit Yield', GEC journal of Research, Vol 2, No 1, pp 1-15, June 1984.

Surveys and analyses virtually all techniques and architectures that use yield enhancing techniques. Evaluates yield models and surmises the overall potential for yield enhancement techniques.

[82] R.D. McKirdy, R.M. Lea, 'Thermal Management for WSI', Technical Memorandum No CM/R/141, Brunel University 1985.

Discusses thermal management issues for WSI. Looks at dissipation limits for different wafer sizes, air flows and

junction temperatures.

[83] R.D. McKirdy, R.M. Lea, 'Physical Design Issues for WSI', Proceedings of an International Workshop on Wafer-scale Integration, Grenoble University, 1986.

Reviews problem areas faced in packaging WSI. Focusses on thermal limits in WSI. Reports on reliability improvements achievable.